Gated RNN & Sequence Generation

Hung-yi Lee
李宏毅

Outline

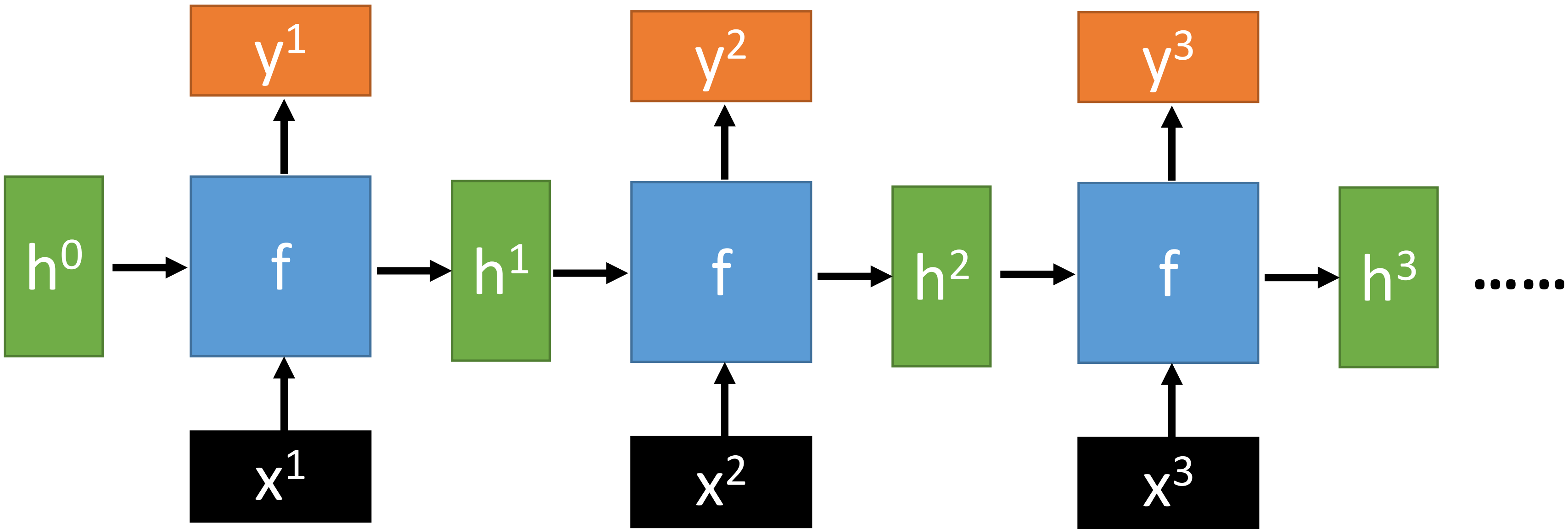
- RNN with Gated Mechanism
- Sequence Generation
- Conditional Sequence Generation
- Tips for Generation

RNN with Gated Mechanism

Recurrent Neural Network

- Given function f: $h', y = f(h, x)$

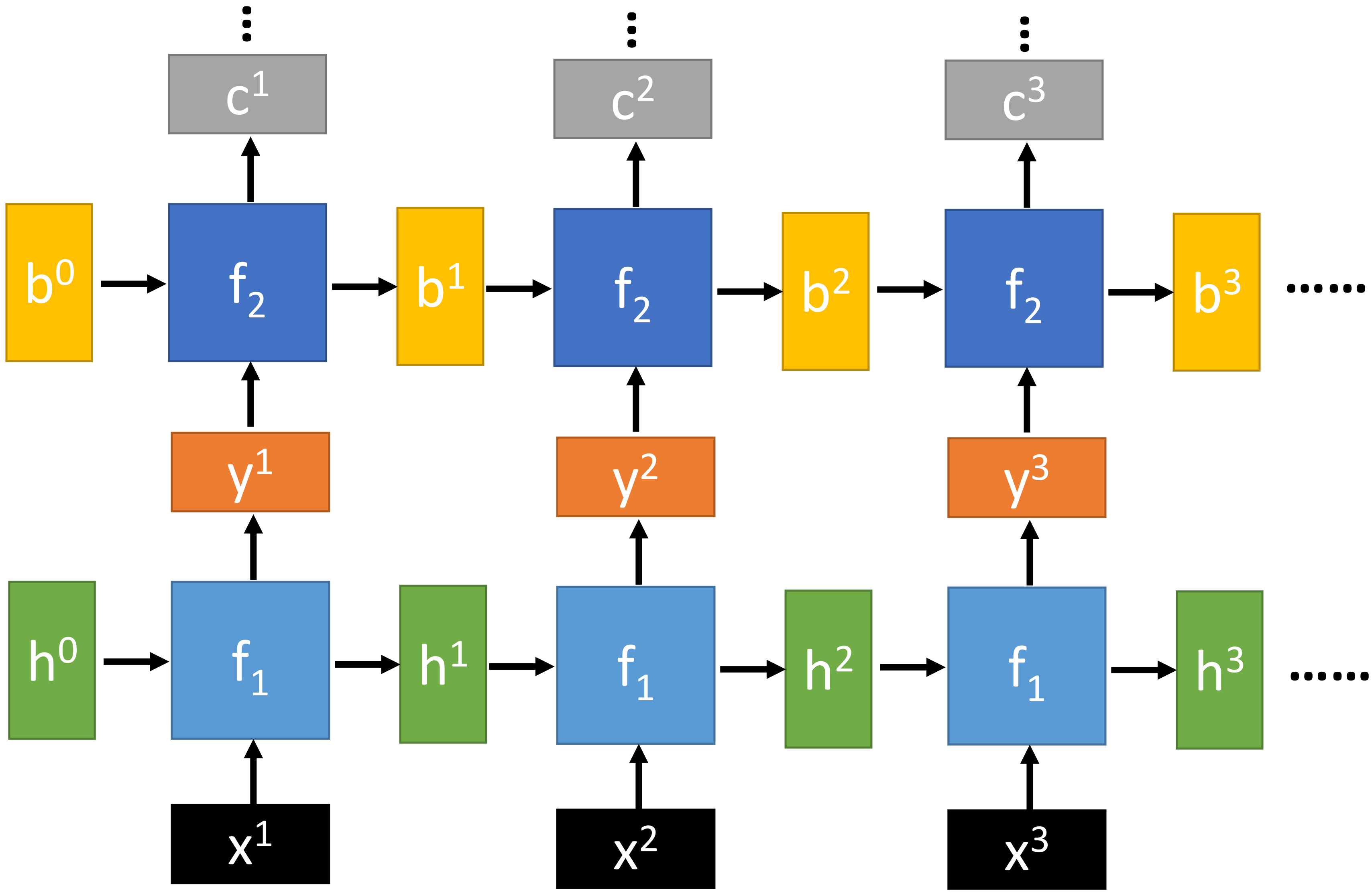
h and h' are vectors with the same dimension



No matter how long the input/output sequence is, we only need one function f

Deep RNN

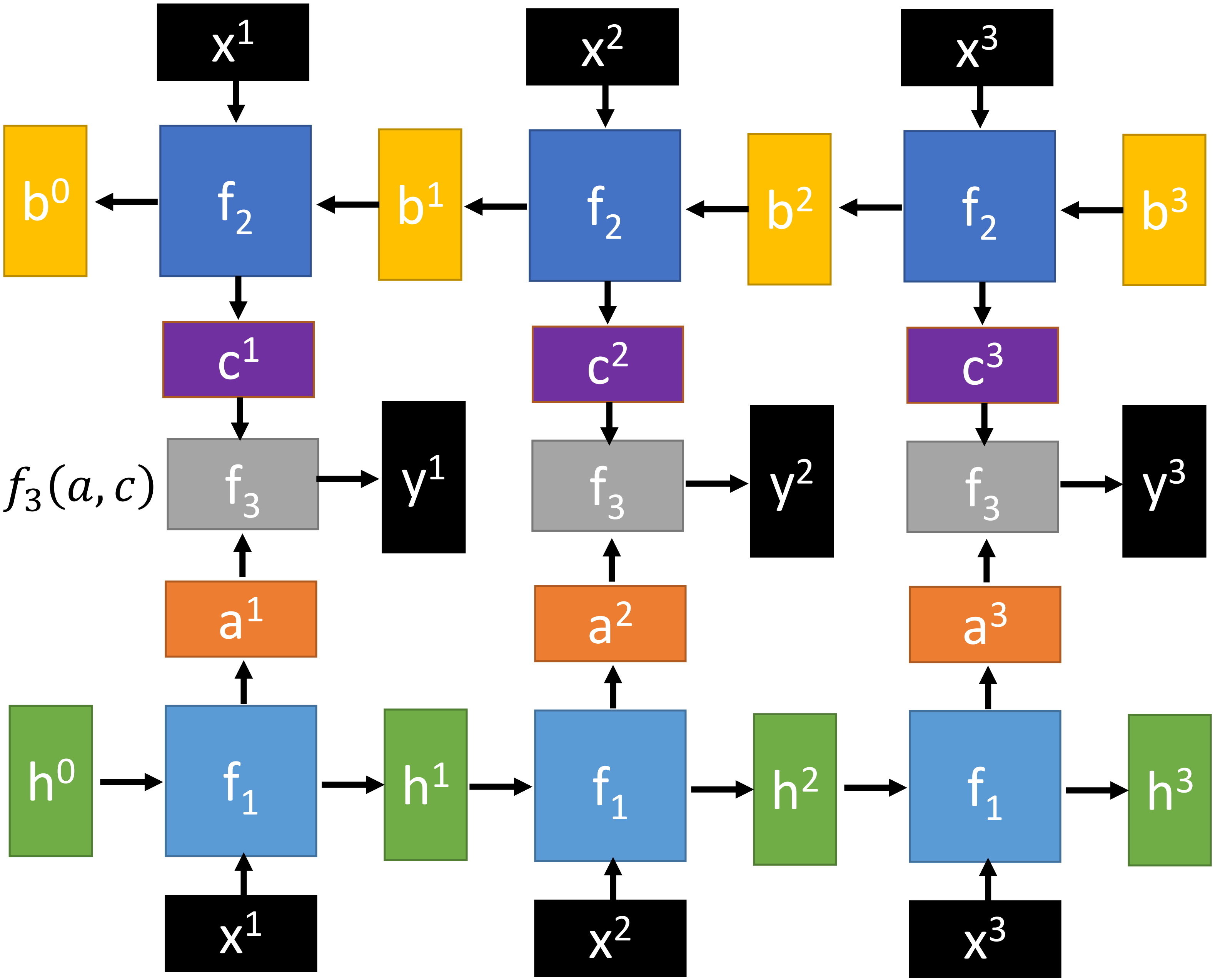
$$h', y = f_1(h, x) \quad b', c = f_2(b, y) \quad \cdots$$



Bidirectional RNN

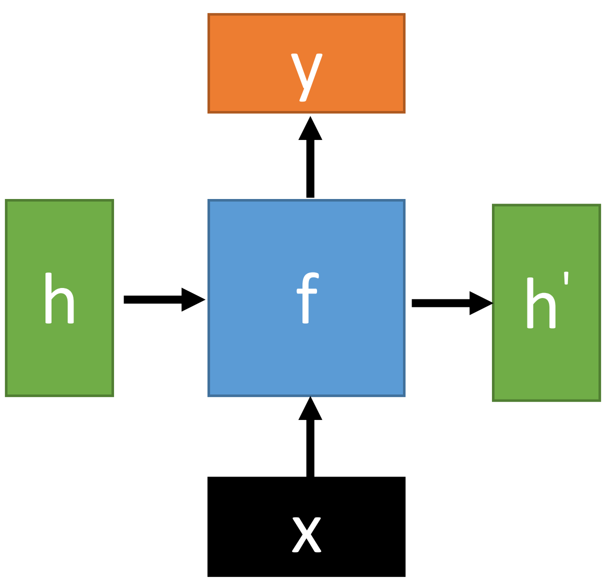
$h', a = f_1(h, x) \quad b', c = f_2(b, x)$

$y = f_3(a, c)$



Naïve RNN

- Given function f: $h', y = f(h, x)$



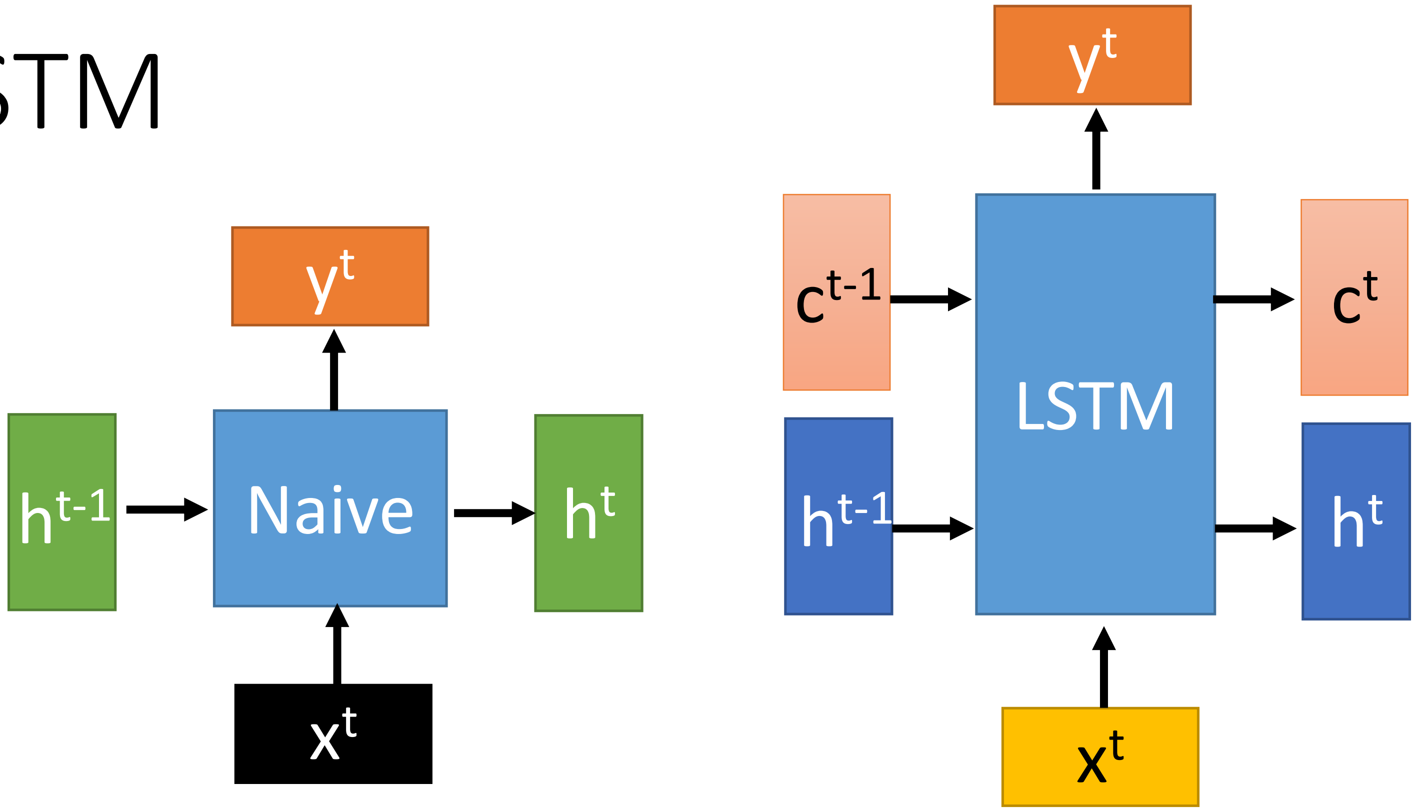
$$h' = \sigma(W^h h + W^i x)$$

$$y = \sigma(W^o h')$$

softmax

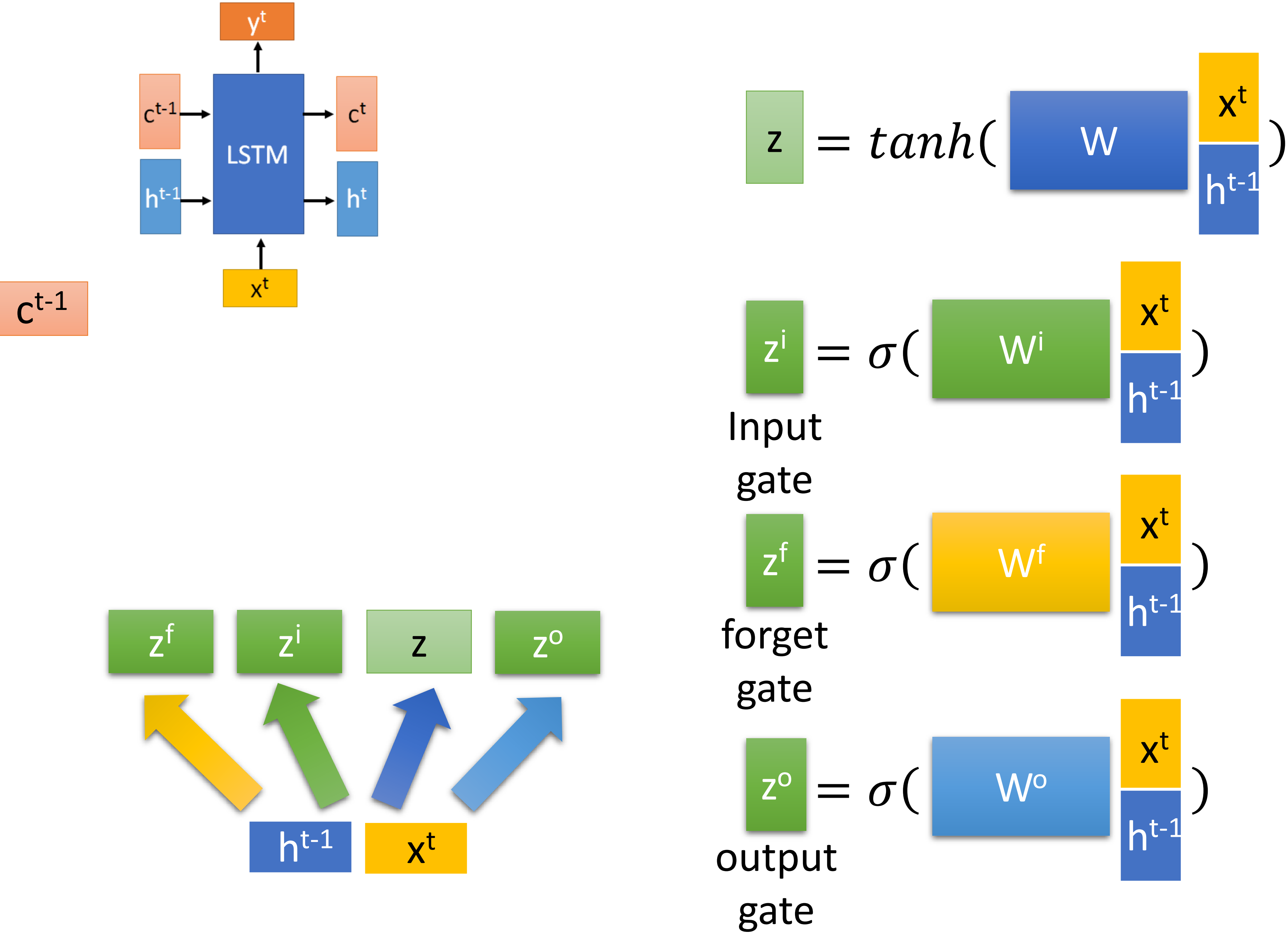
Ignore bias here

LSTM

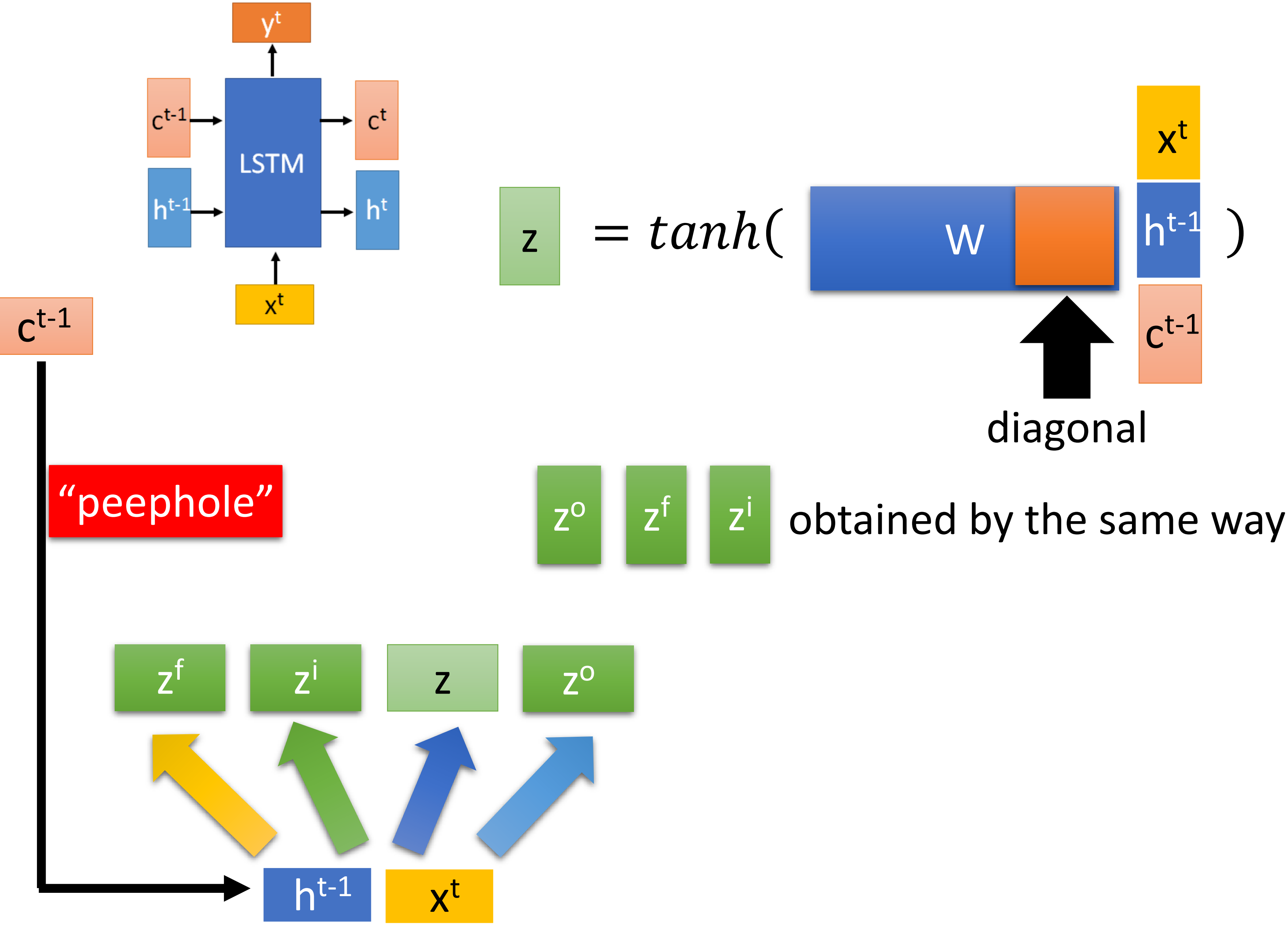


c changes slowly ➡ c^t is c^{t-1} added by something

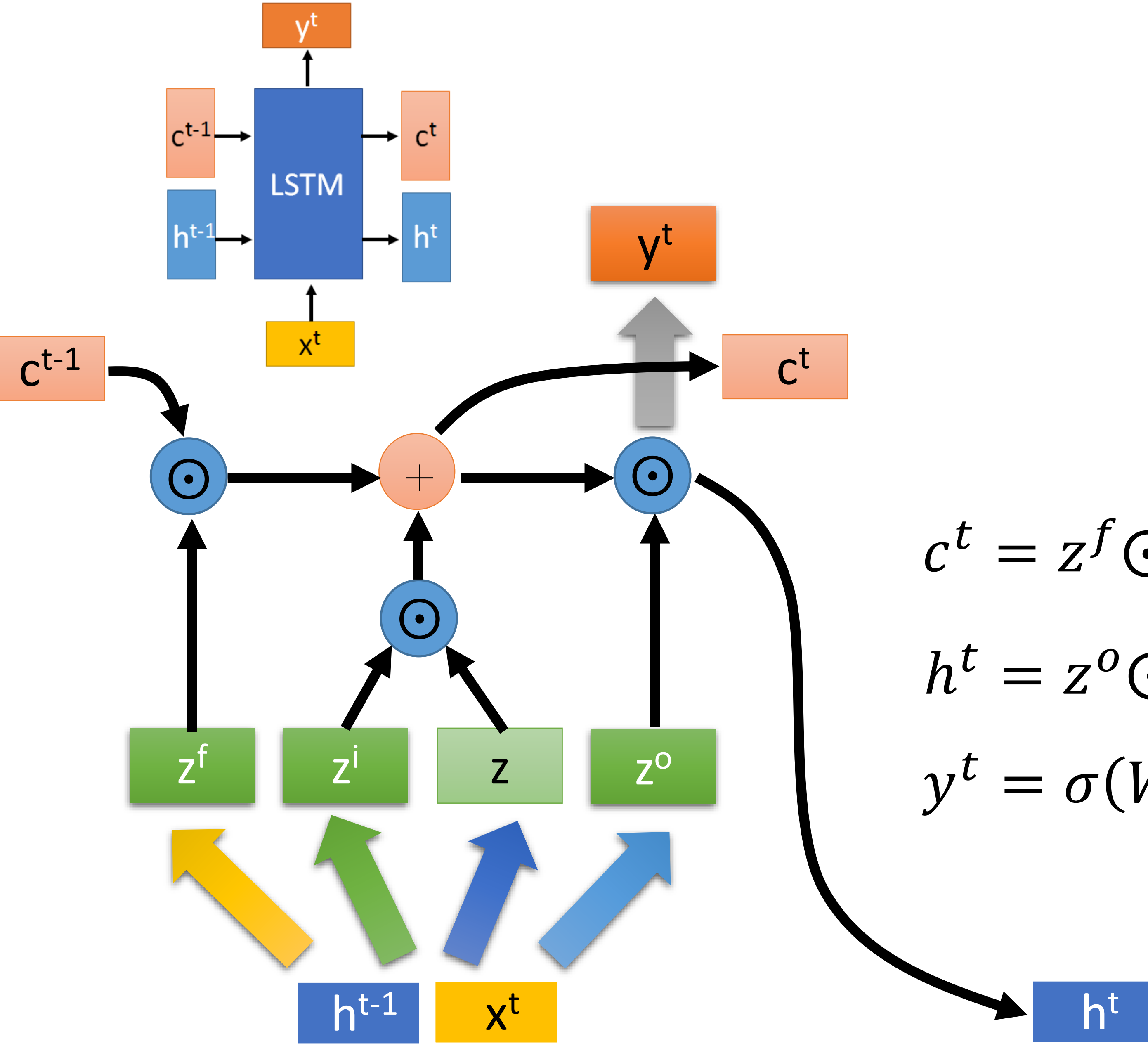
h changes faster ➡ h^t and h^{t-1} can be very different



y^t
c^{t-1}
LSTM
c^t
h^{t-1}
h^t
x^t
c^{t-1}
$z = tanh($ W x^t h^{t-1} $)$
$z^i = \sigma($ W^i x^t h^{t-1} $)$
Input gate
$z^f = \sigma($ W^f x^t h^{t-1} $)$
forget gate
$z^o = \sigma($ W^o x^t h^{t-1} $)$
output gate
z^f
z^i
z
z^o
h^{t-1}
x^t



y^t
c^t-1
LSTM
c^t
h^t-1
h^t
x^t
z
= tanh(
W
x^t
h^t-1
)
c^t-1
diagonal
c^t-1
“peephole”
z^o
z^f
z^i
obtained by the same way
z^f
z^i
z
z^o
h^t-1
x^t

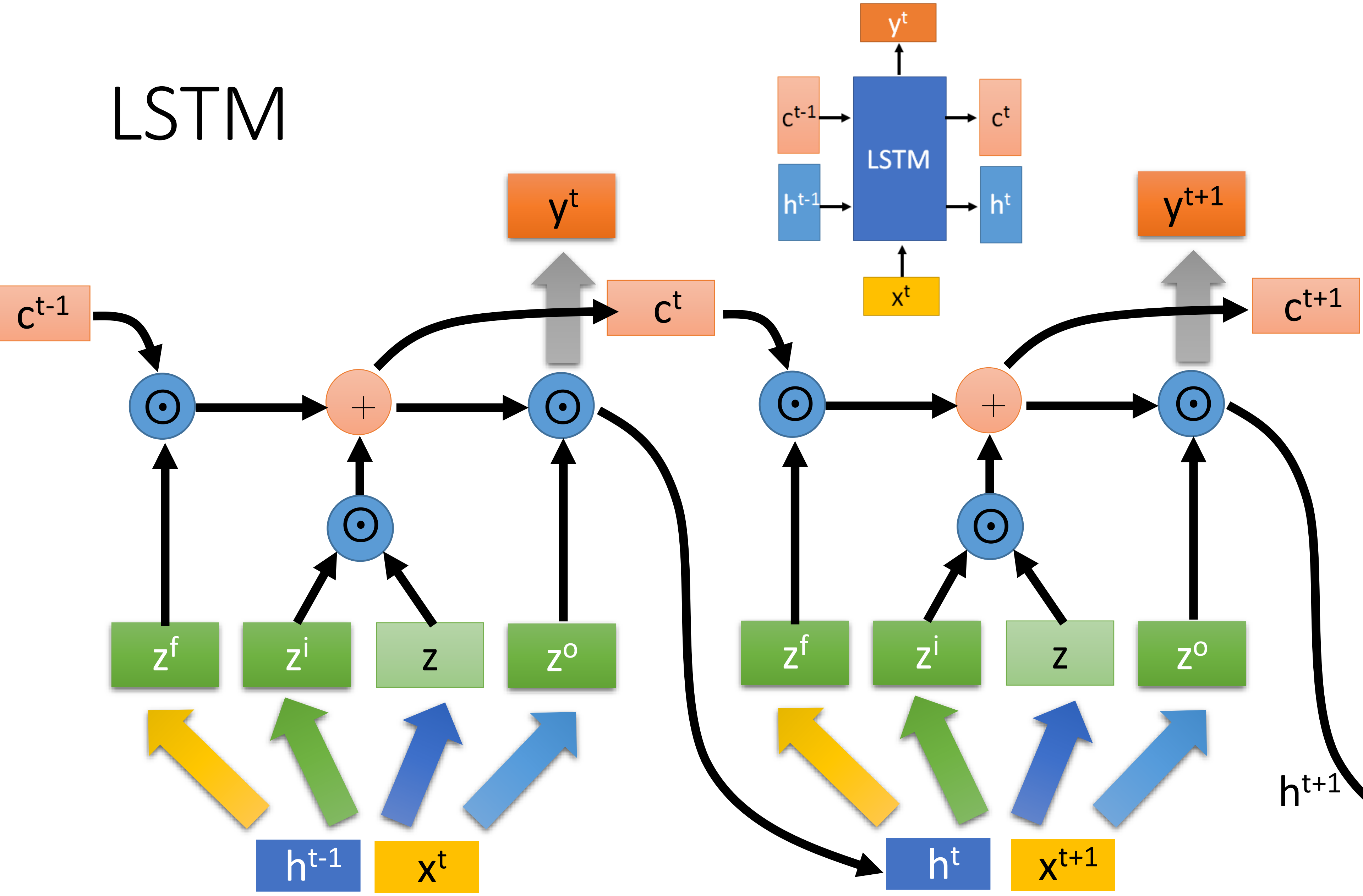


$$c^t = z^f \odot c^{t-1} + z^i \odot z$$

$$h^t = z^o \odot tanh(c^t)$$

$$y^t = \sigma(W' h^t)$$

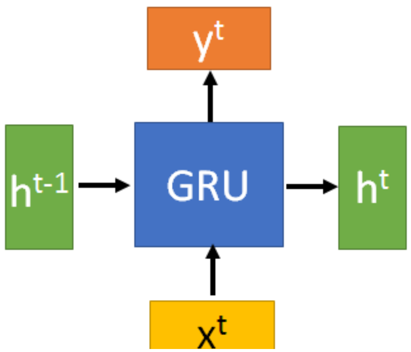
LSTM

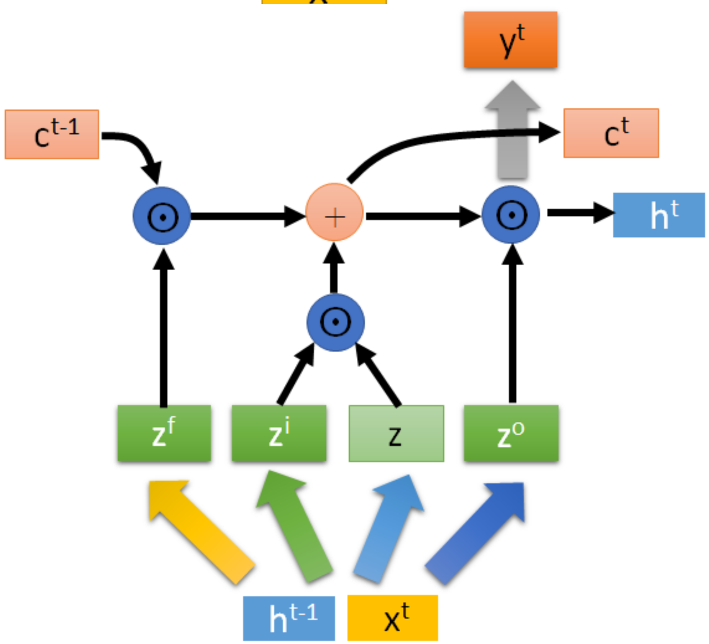


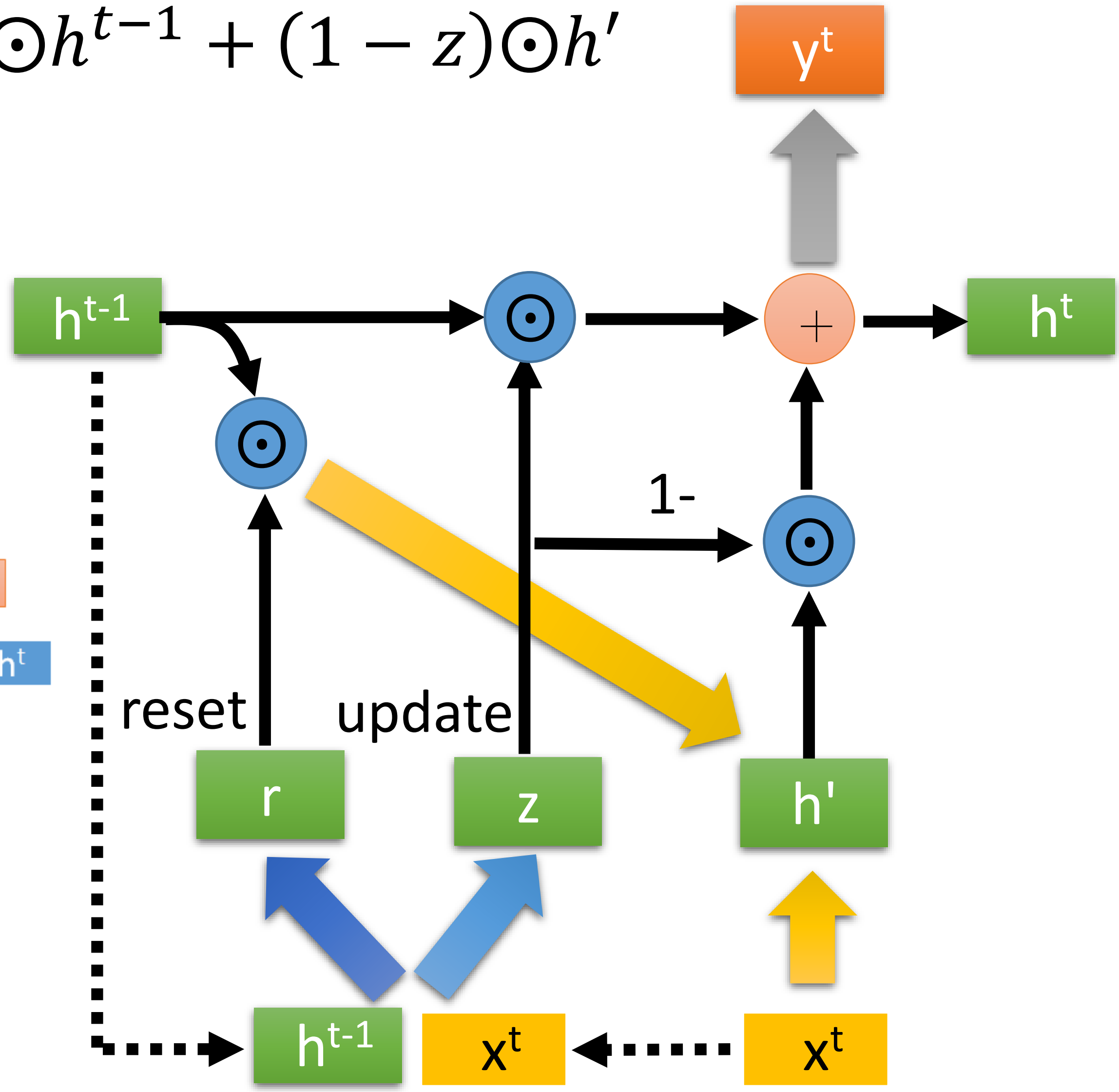
yt
ct-1
LSTM
ct
ht-1
ht
xt
yt
ct-1
ct
zf
zi
z
zo
ht-1
xt
yt+1
ct+1
zf
zi
z
zo
ht
xt+1
ht+1

GRU

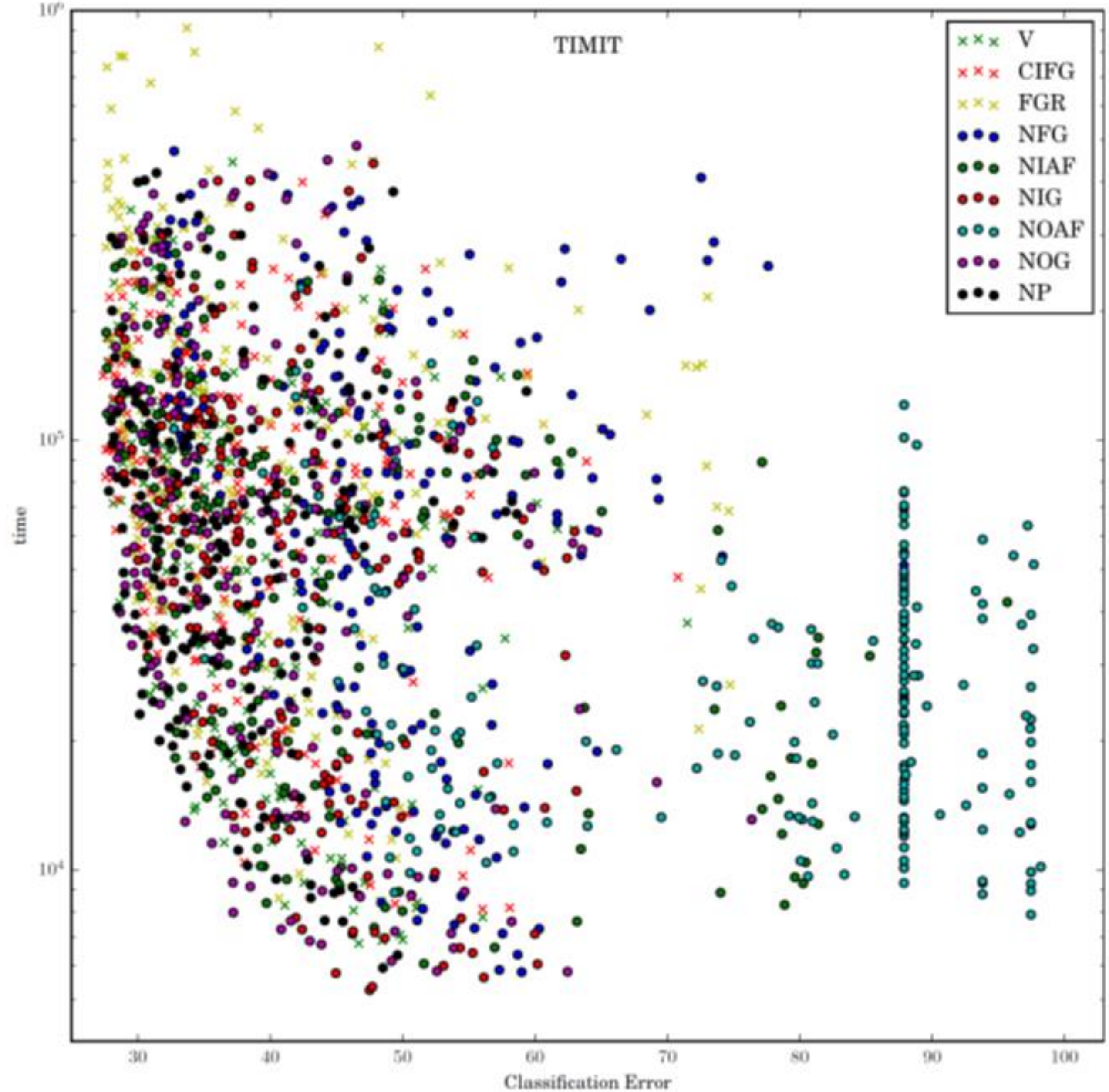
$$h^t = z \odot h^{t-1} + (1-z) \odot h'$$





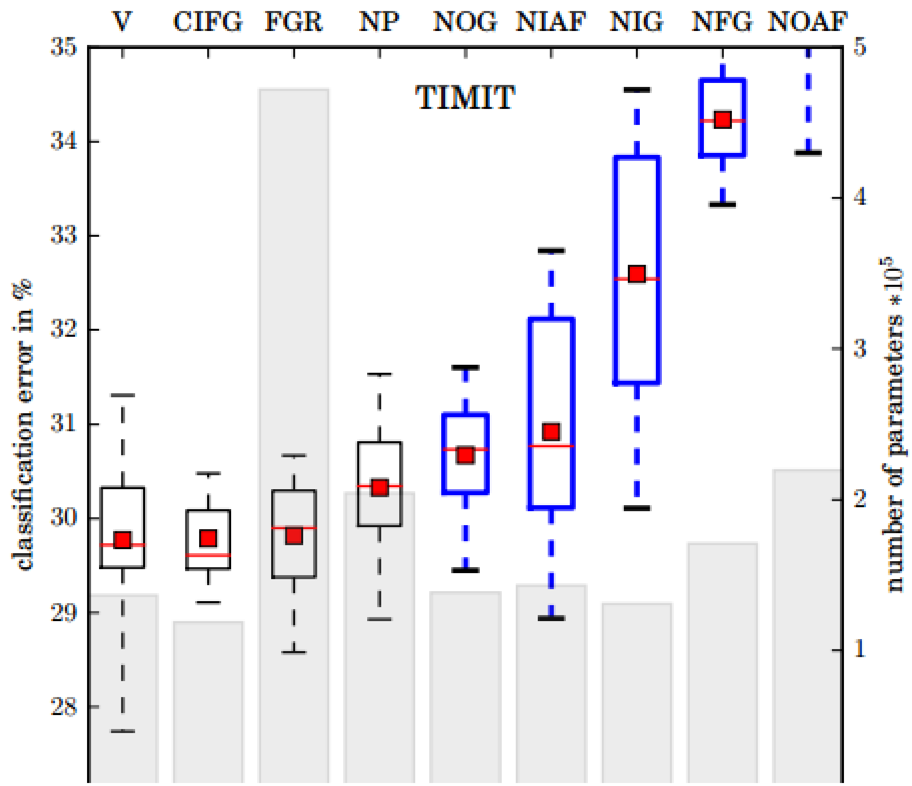


LSTM: A Search Space Odyssey



LSTM: A Search Space Odyssey

1. No Input Gate (NIG)
2. No Forget Gate (NFG)
3. No Output Gate (NOG)
4. No Input Activation Function (NIAF)
5. No Output Activation Function (NOAF)
6. No Peepholes (NP)
7. Coupled Input and Forget Gate (CIFG)
8. Full Gate Recurrence (FGR)



Standard LSTM works well

Simply LSTM: coupling input and forget gate, removing peephole

Forget gate is critical for performance

Output gate activation function is critical

An Empirical Exploration of Recurrent Network Architectures

Arch.	Arith.	XML	PTB
Tanh	0.29493	0.32050	0.08782
LSTM	0.89228	0.42470	0.08912
LSTM-f	0.29292	0.23356	0.08808
LSTM-i	0.75109	0.41371	0.08662
LSTM-o	0.86747	0.42117	0.08933
LSTM-b	0.90163	0.44434	0.08952
GRU	0.89565	0.45963	0.09069
MUT1	**0.92135**	**0.47483**	0.08968
MUT2	0.89735	**0.47324**	0.09036
MUT3	0.90728	0.46478	**0.09161**

LSTM-f/i/o: removing forget/input/output gates

LSTM-b: large bias

Importance: forget > input > output

Large bias for forget gate is helpful

An Empirical Exploration of Recurrent Network Architectures

MUT1:

$$
\begin{aligned}
z &= \mathrm{sigm}(W_{\mathrm{xz}} x_t + b_{\mathrm{z}}) \\
r &= \mathrm{sigm}(W_{\mathrm{xr}} x_t + W_{\mathrm{hr}} h_t + b_{\mathrm{r}}) \\
h_{t+1} &= \tanh(W_{\mathrm{hh}}(r \odot h_t) + \tanh(x_t) + b_{\mathrm{h}}) \odot z \\
&+ h_t \odot (1 - z)
\end{aligned}
$$

MUT2:

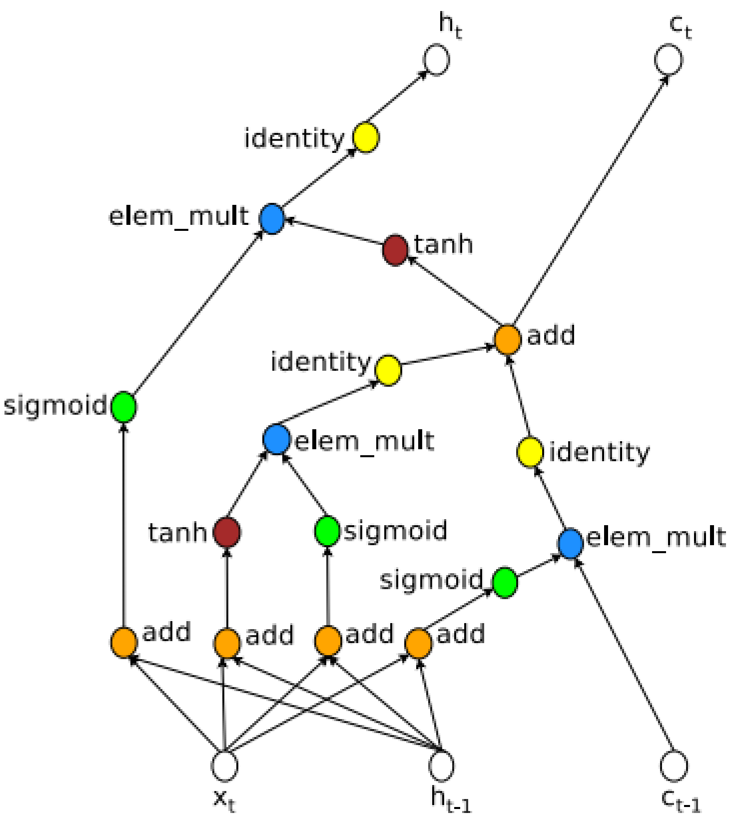
$$
\begin{aligned}
z &= \mathrm{sigm}(W_{\mathrm{xz}} x_t + W_{\mathrm{hz}} h_t + b_{\mathrm{z}}) \\
r &= \mathrm{sigm}(x_t + W_{\mathrm{hr}} h_t + b_{\mathrm{r}}) \\
h_{t+1} &= \tanh(W_{\mathrm{hh}}(r \odot h_t) + W_{xh} x_t + b_{\mathrm{h}}) \odot z \\
&+ h_t \odot (1 - z)
\end{aligned}
$$

MUT3:

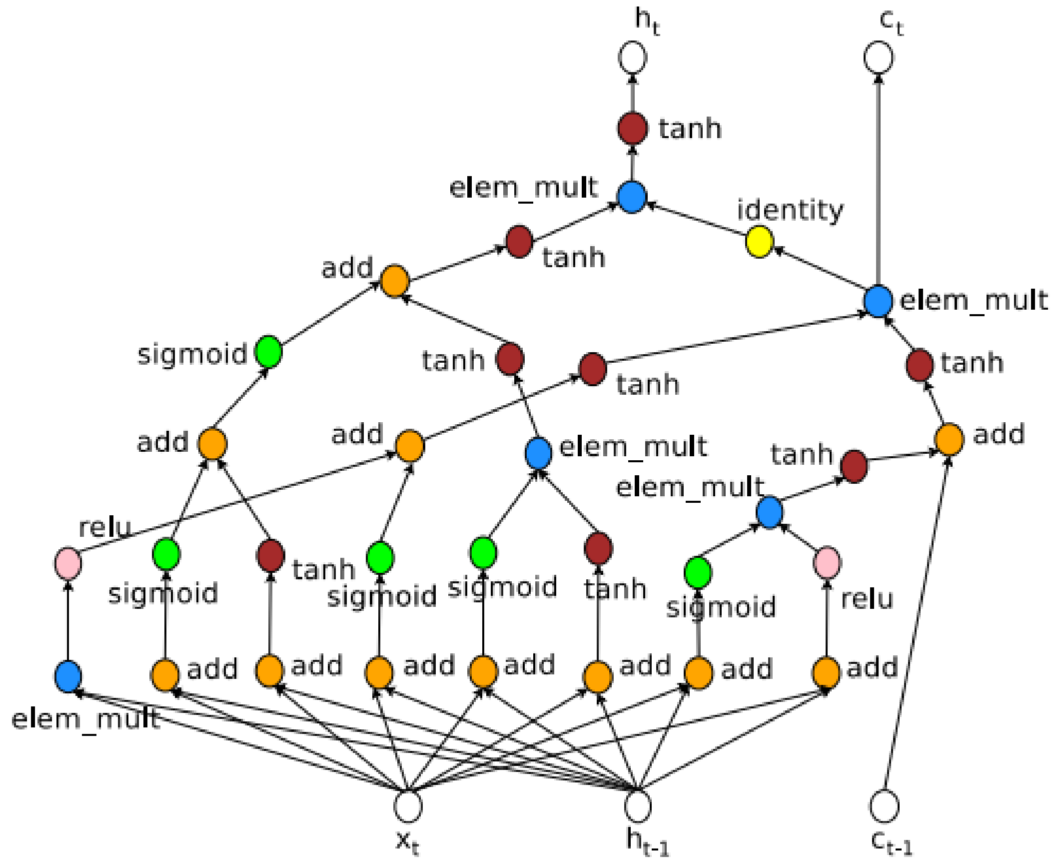
$$
\begin{aligned}
z &= \mathrm{sigm}(W_{\mathrm{xz}} x_t + W_{\mathrm{hz}} \tanh(h_t) + b_{\mathrm{z}}) \\
r &= \mathrm{sigm}(W_{\mathrm{xr}} x_t + W_{\mathrm{hr}} h_t + b_{\mathrm{r}}) \\
h_{t+1} &= \tanh(W_{\mathrm{hh}}(r \odot h_t) + W_{xh} x_t + b_{\mathrm{h}}) \odot z \\
&+ h_t \odot (1 - z)
\end{aligned}
$$

Neural Architecture Search with Reinforcement Learning

LSTM

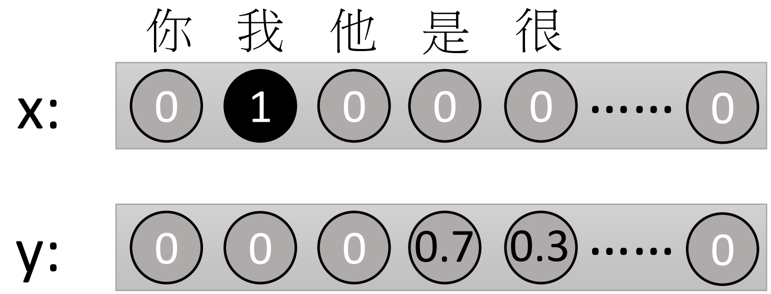


From Reinforcement Learning

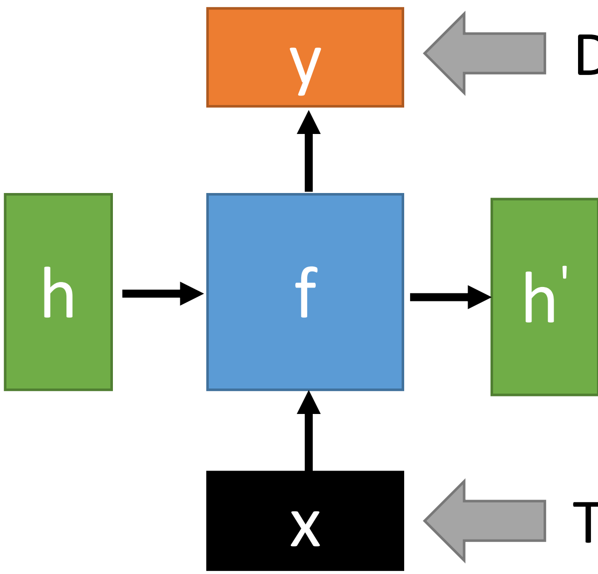


Sequence Generation

Generation



- Sentences are composed of characters/words
- Generating a character/word at each time by RNN



Distribution over the token

(sampling from the distribution to generate a token)

The token generated at the last time step.

(represented by 1-of-N encoding)

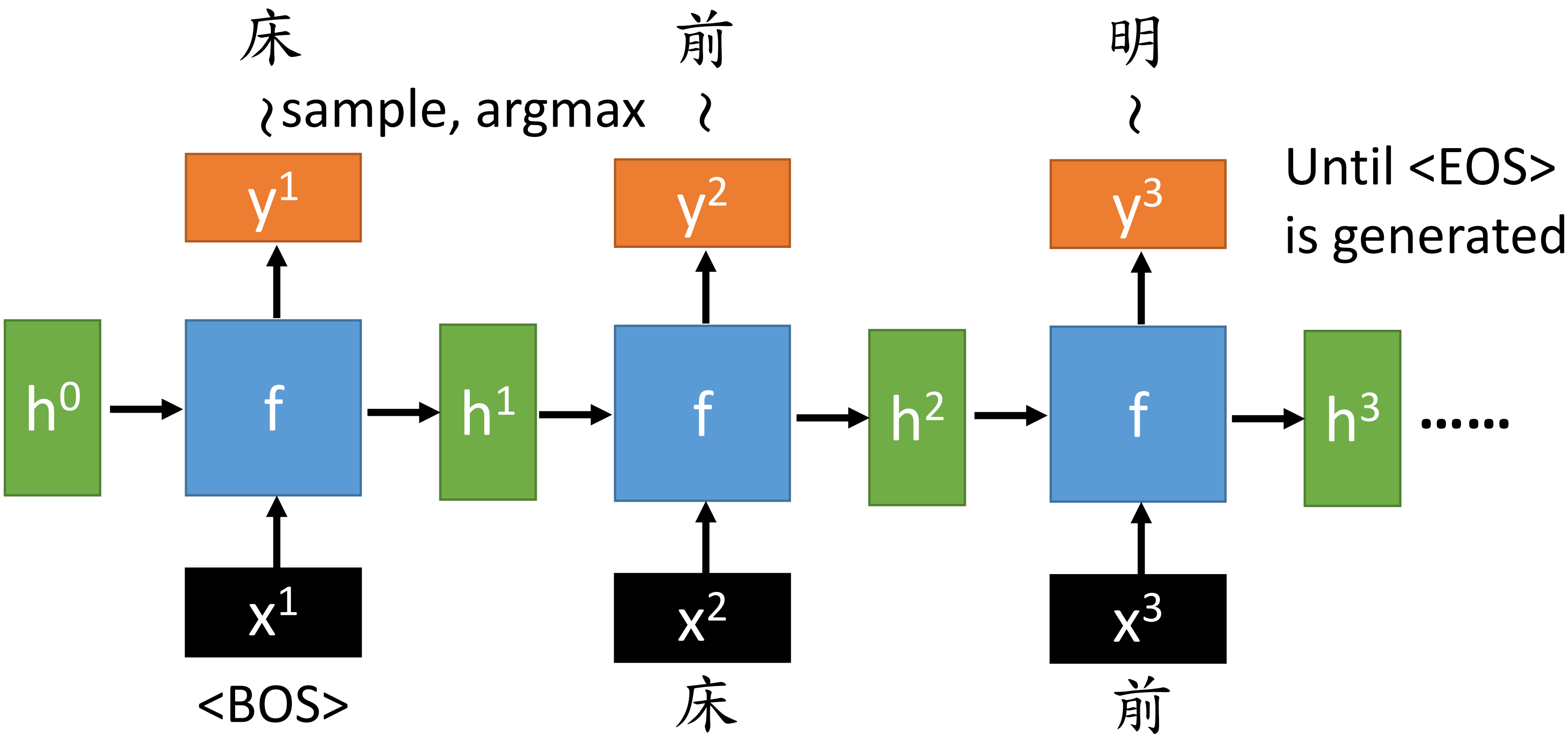
Generation

y^1: P(w|<BOS>)

y^2: P(w|<BOS>,床)

y^3: P(w|<BOS>,床,前)

- Sentences are composed of characters/words
- Generating a character/word at each time by RNN

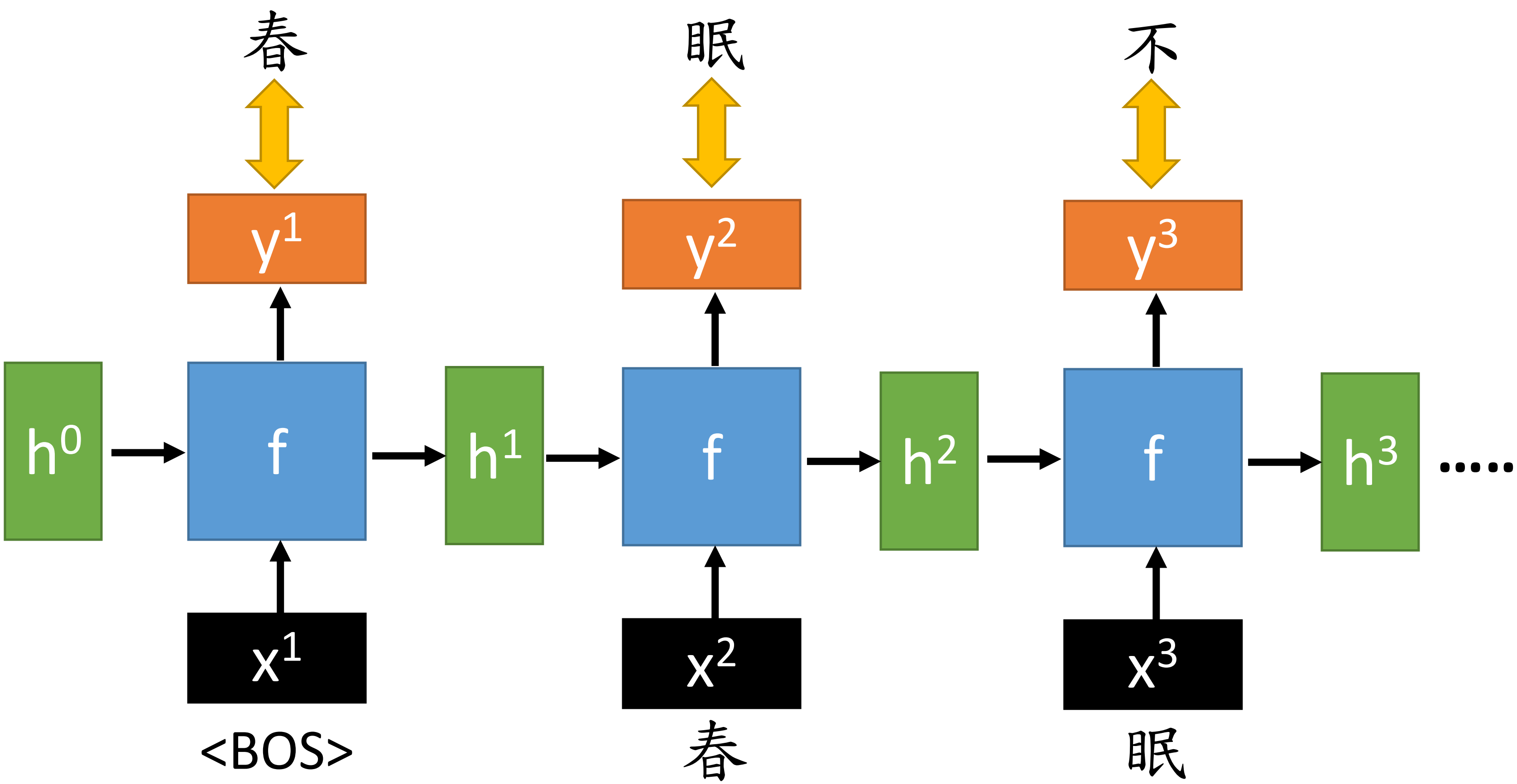


Generation

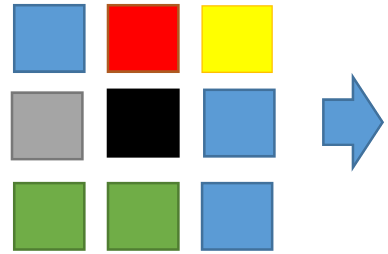
: minimizing cross-entropy

- ***Training***

Training data: 春 眠 不 覺 曉



Generation

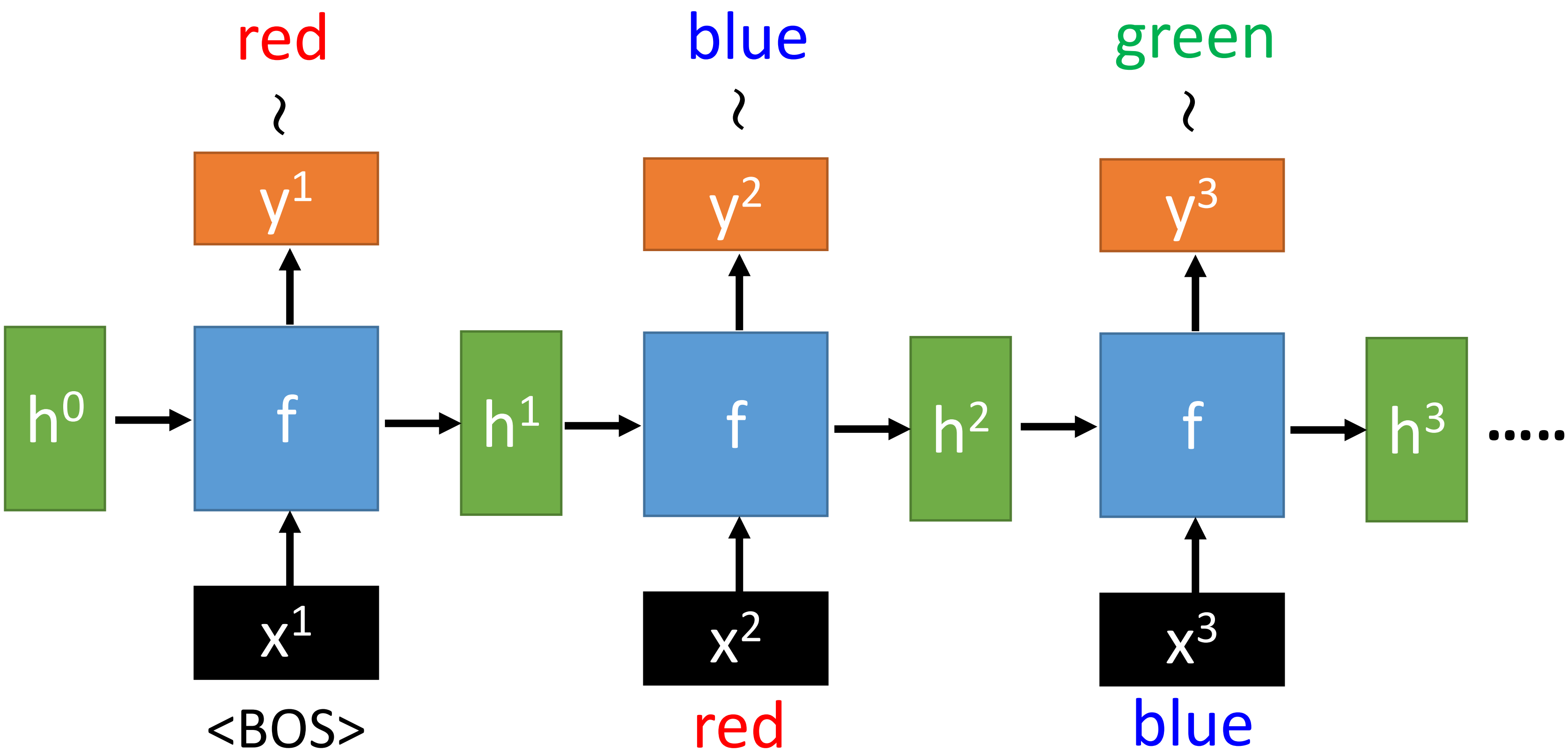


Consider as a sentence

blue red yellow gray ……

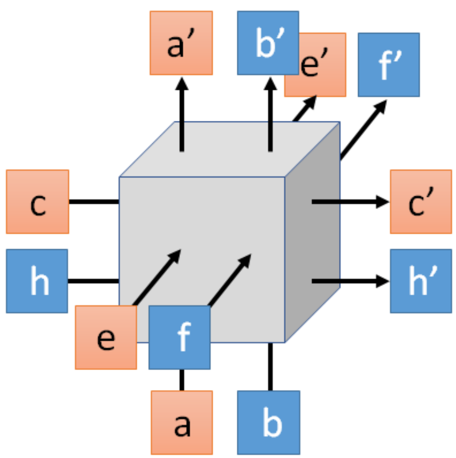
Train a RNN based on the "sentences"

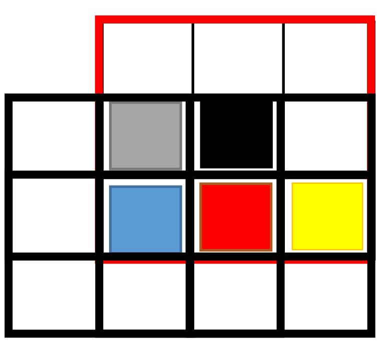
- Images are composed of pixels
- Generating a pixel at each time by RNN

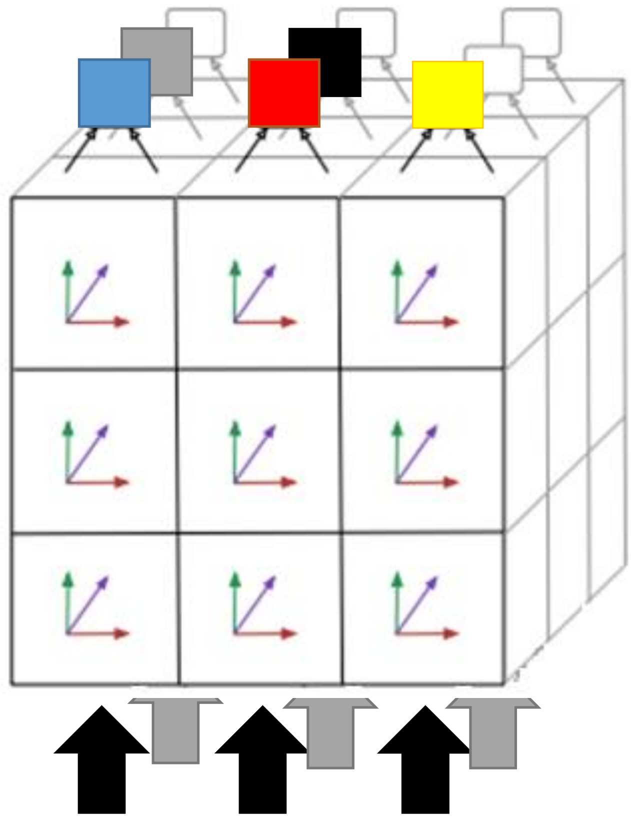


Generation - PixelRNN

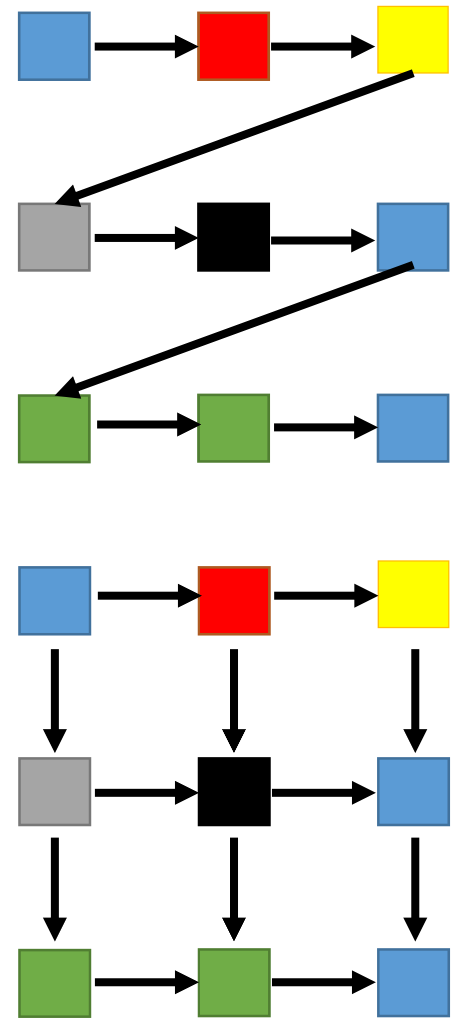
- Images are composed of pixels







3 x 3 images



Conditional Sequence Generation

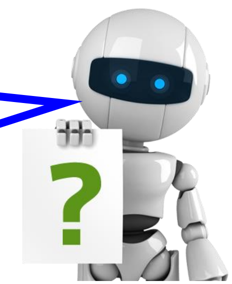
Conditional Generation

- We don’t want to simply generate some random sentences.
- Generate sentences based on conditions:

Caption Generation

Given condition:

“A young girl is dancing.”

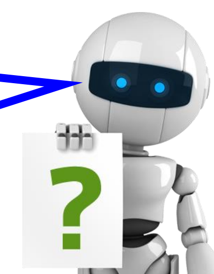


Chat-bot

Given condition:

“Hello”

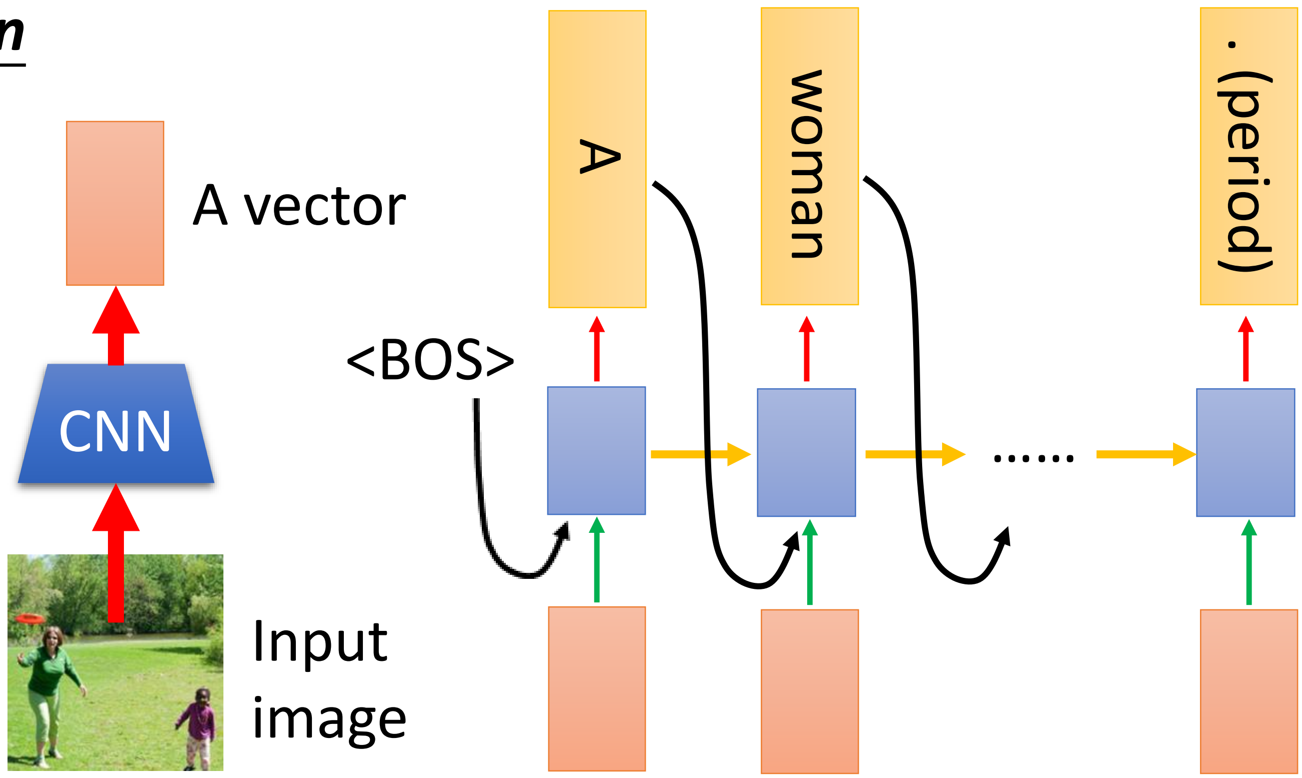
“Hello. Nice to see you.”



Conditional Generation

- Represent the input condition as a vector, and consider the vector as the input of RNN generator

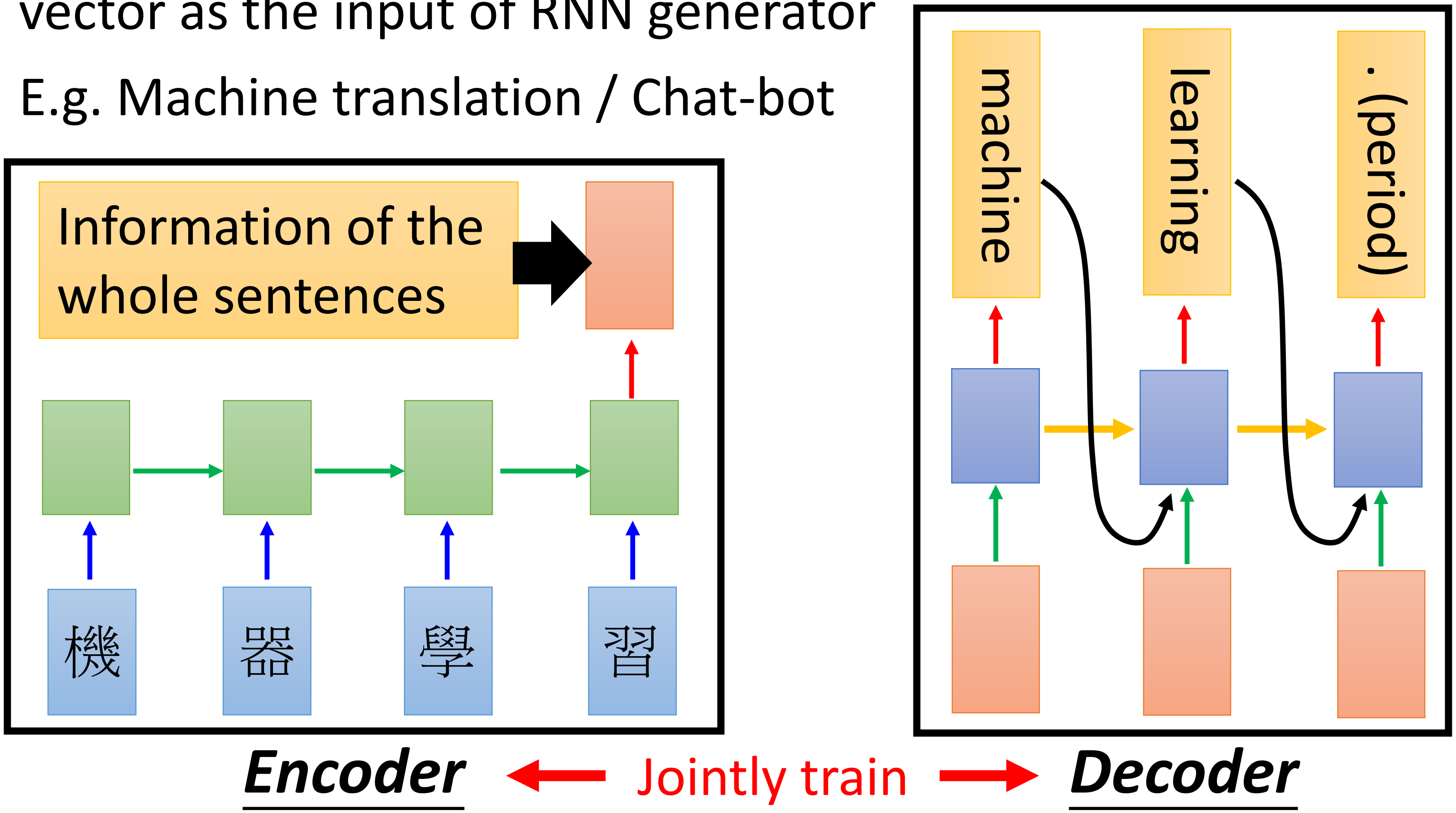
Image Caption Generation



Conditional Generation

Sequence-to-sequence learning

- Represent the input condition as a vector, and consider the vector as the input of RNN generator
- E.g. Machine translation / Chat-bot



Conditional Generation

M: Hello

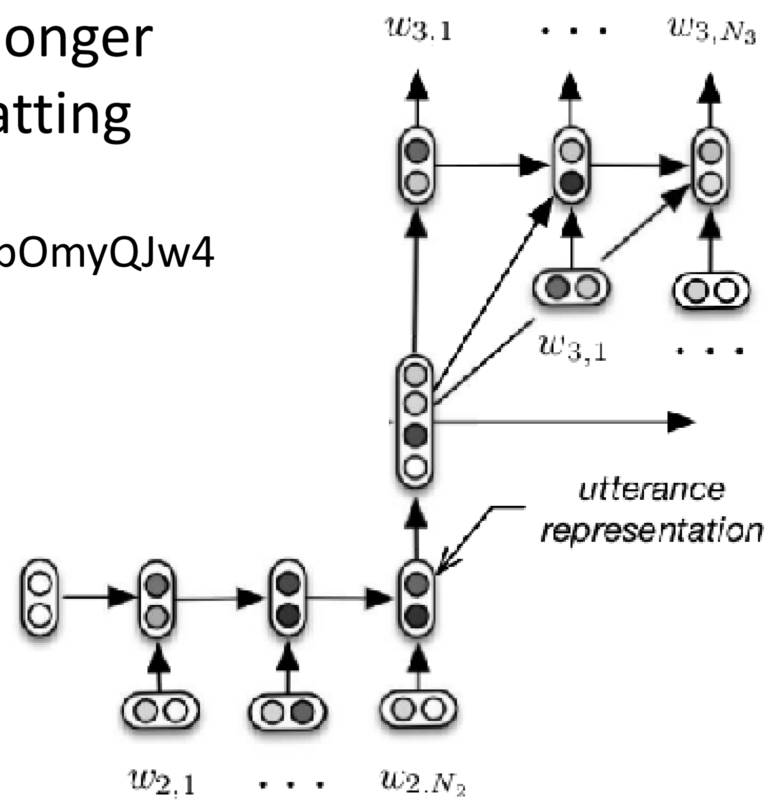
U: Hi

M: Hi

Need to consider longer context during chatting

https://www.youtube.com/watch?v=e2MpOmyQJw4

M: Hi

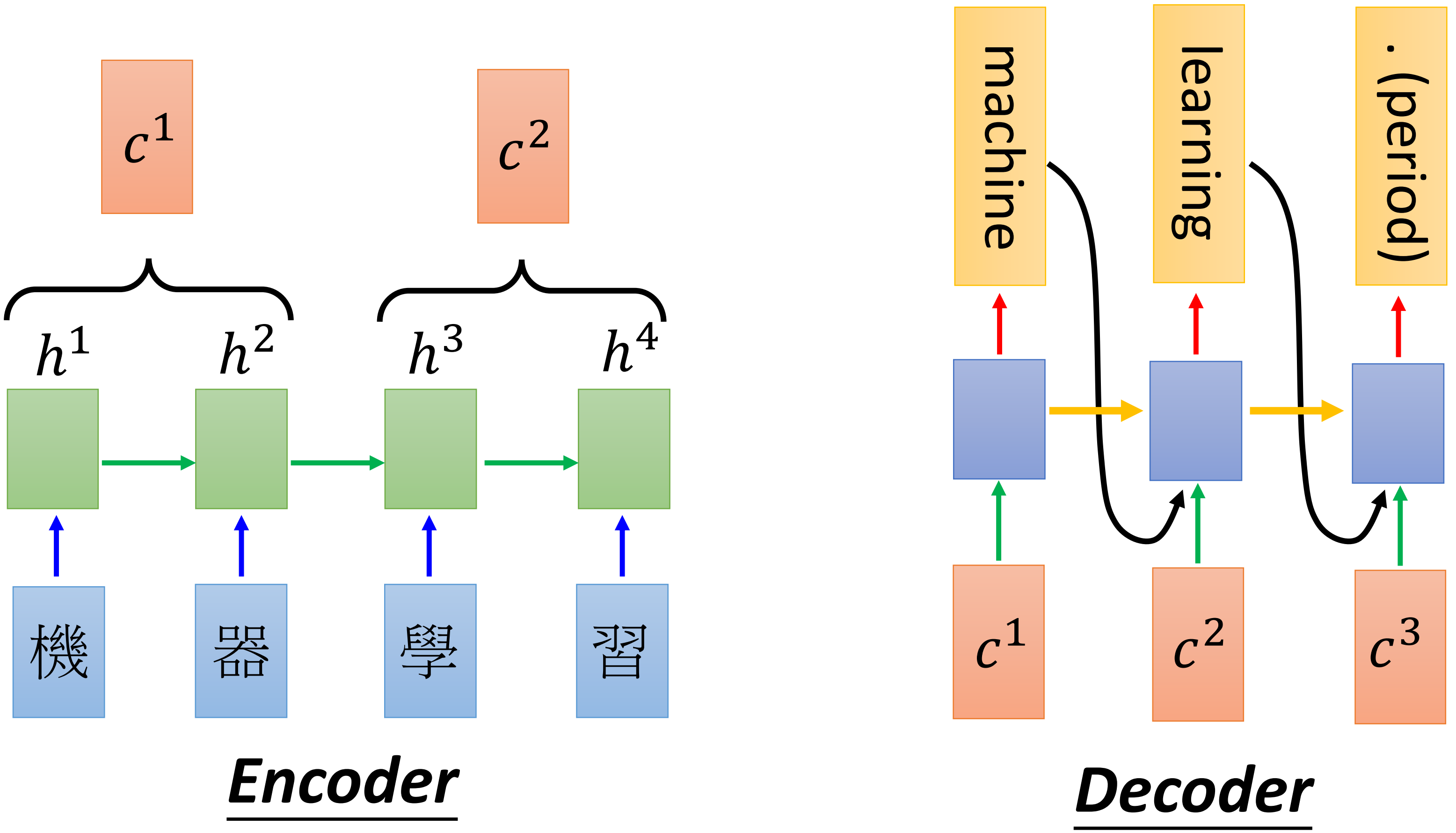


M: Hello

U: Hi

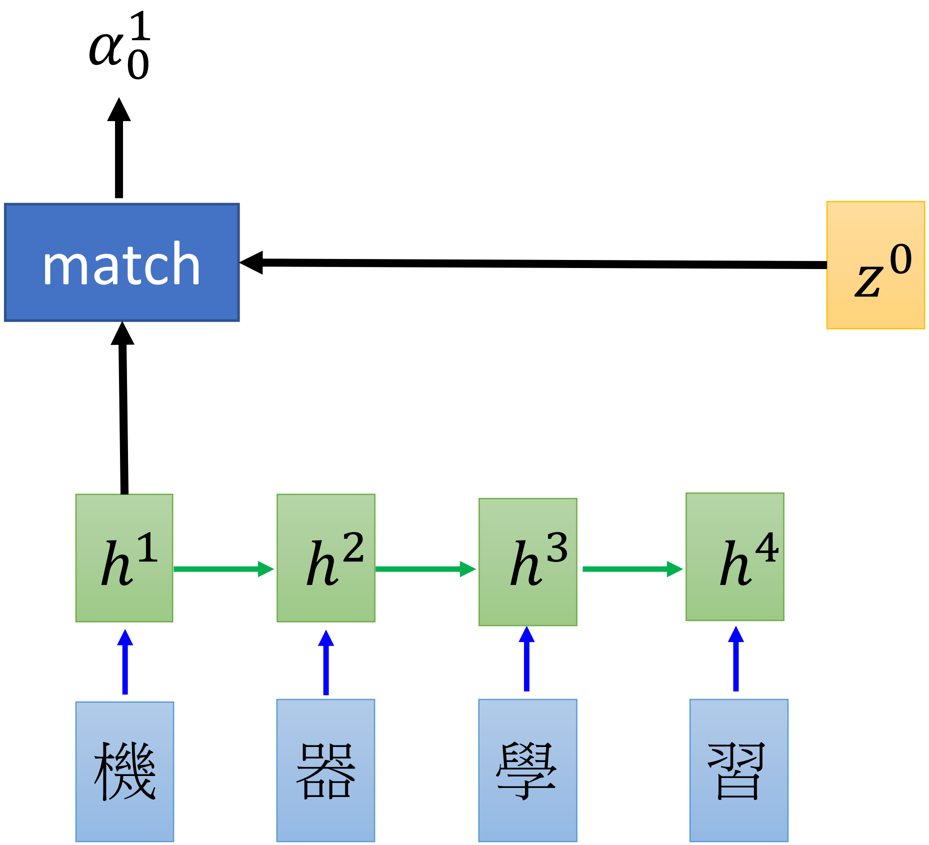
Serban, Iulian V., Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau, 2015 "Building End-To-End Dialogue Systems Using Generative Hierarchical Neural Network Models.

Dynamic Conditional Generation

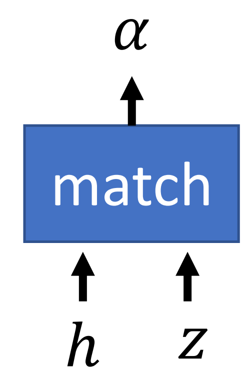


Machine Translation

- Attention-based model



Jointly learned with other part of the network



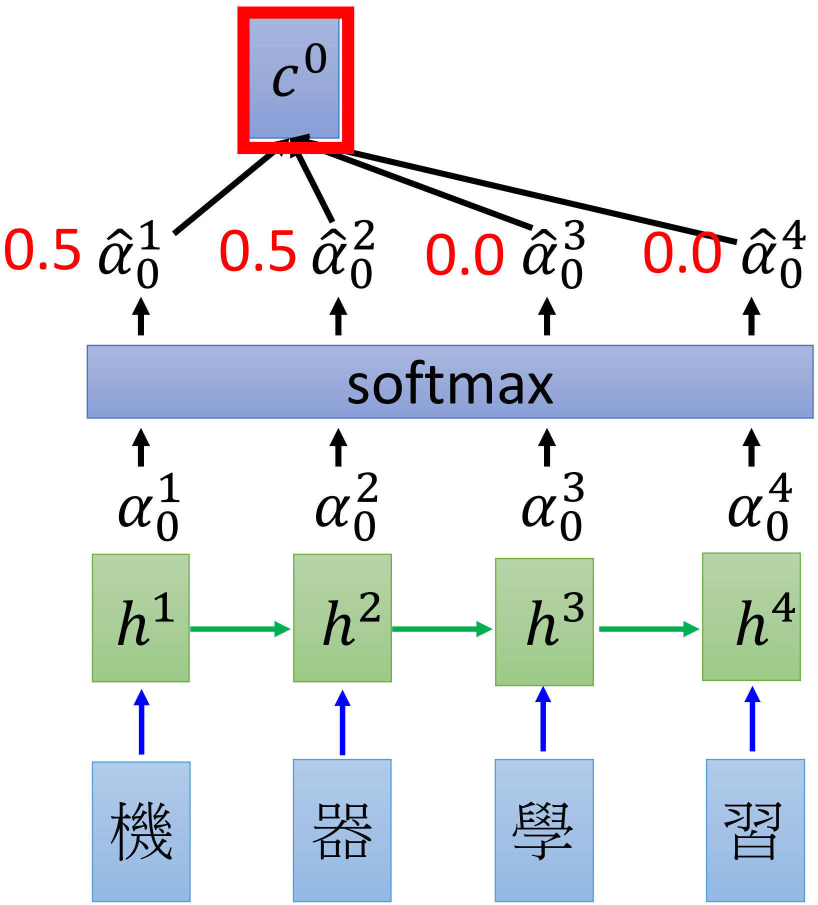
What is match ?

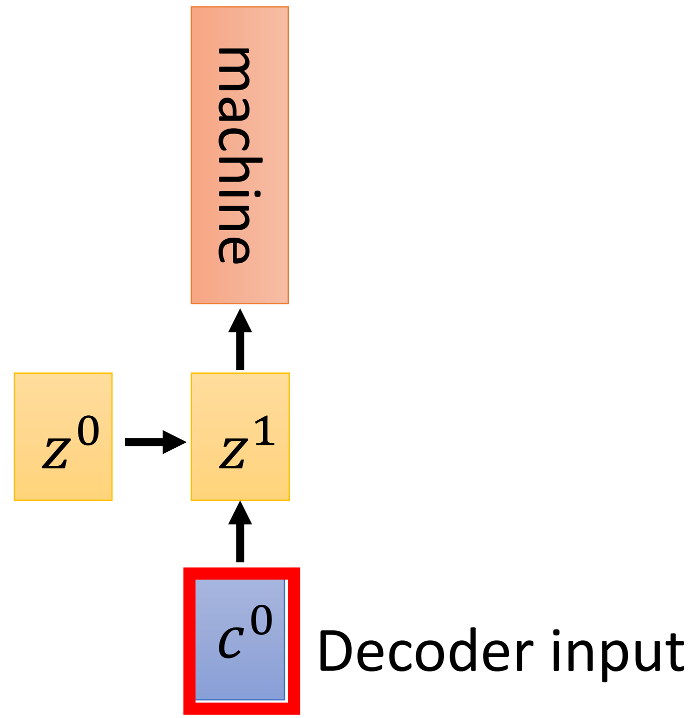
Design by yourself

- ➢ Cosine similarity of z and h
- ➢ Small NN whose input is z and h, output a scalar
- ➢ $\alpha = h^T W z$

Machine Translation

- Attention-based model



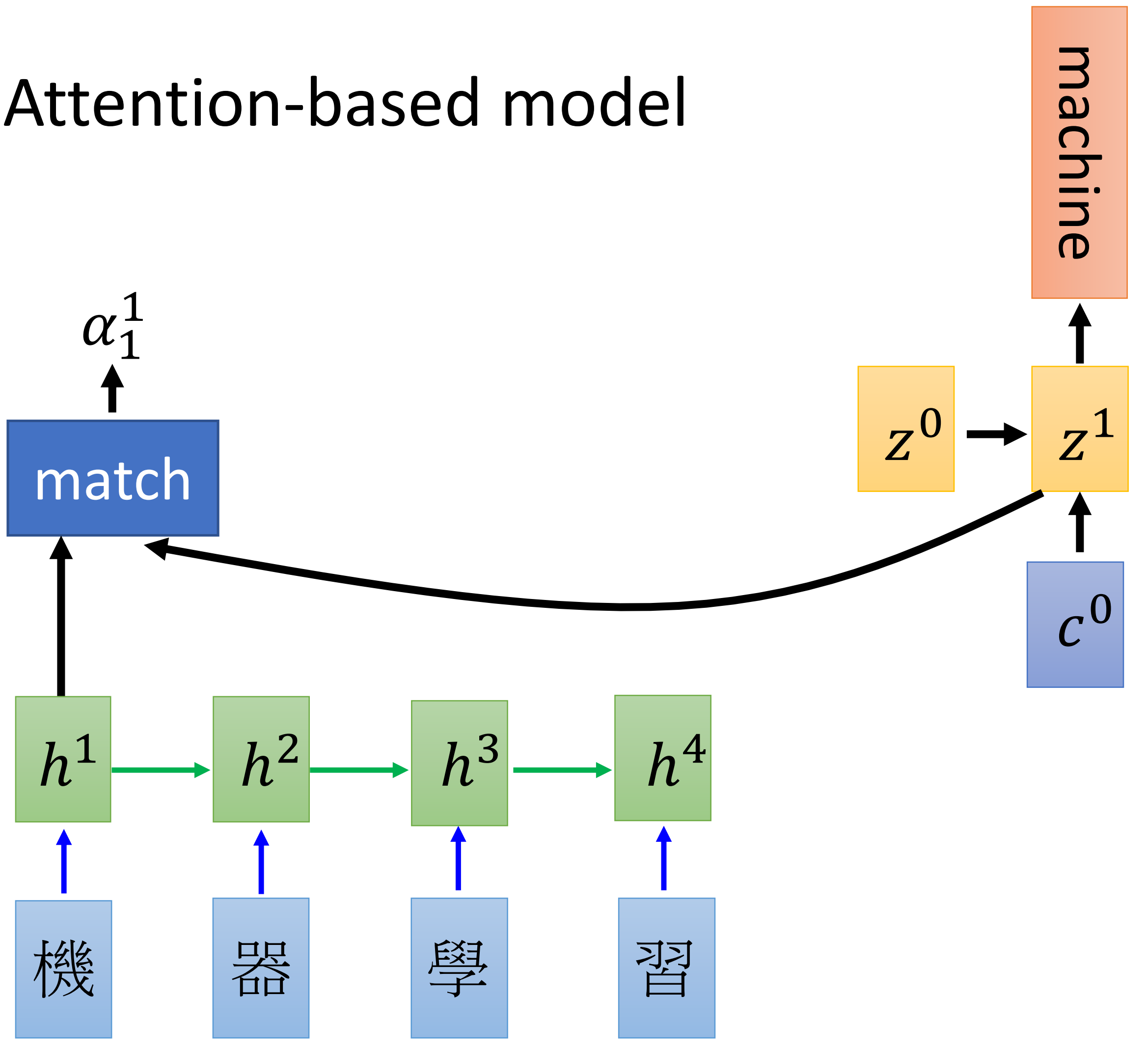


$$c^0 = \sum \hat{\alpha}_0^i h^i$$

$$= 0.5h^1 + 0.5h^2$$

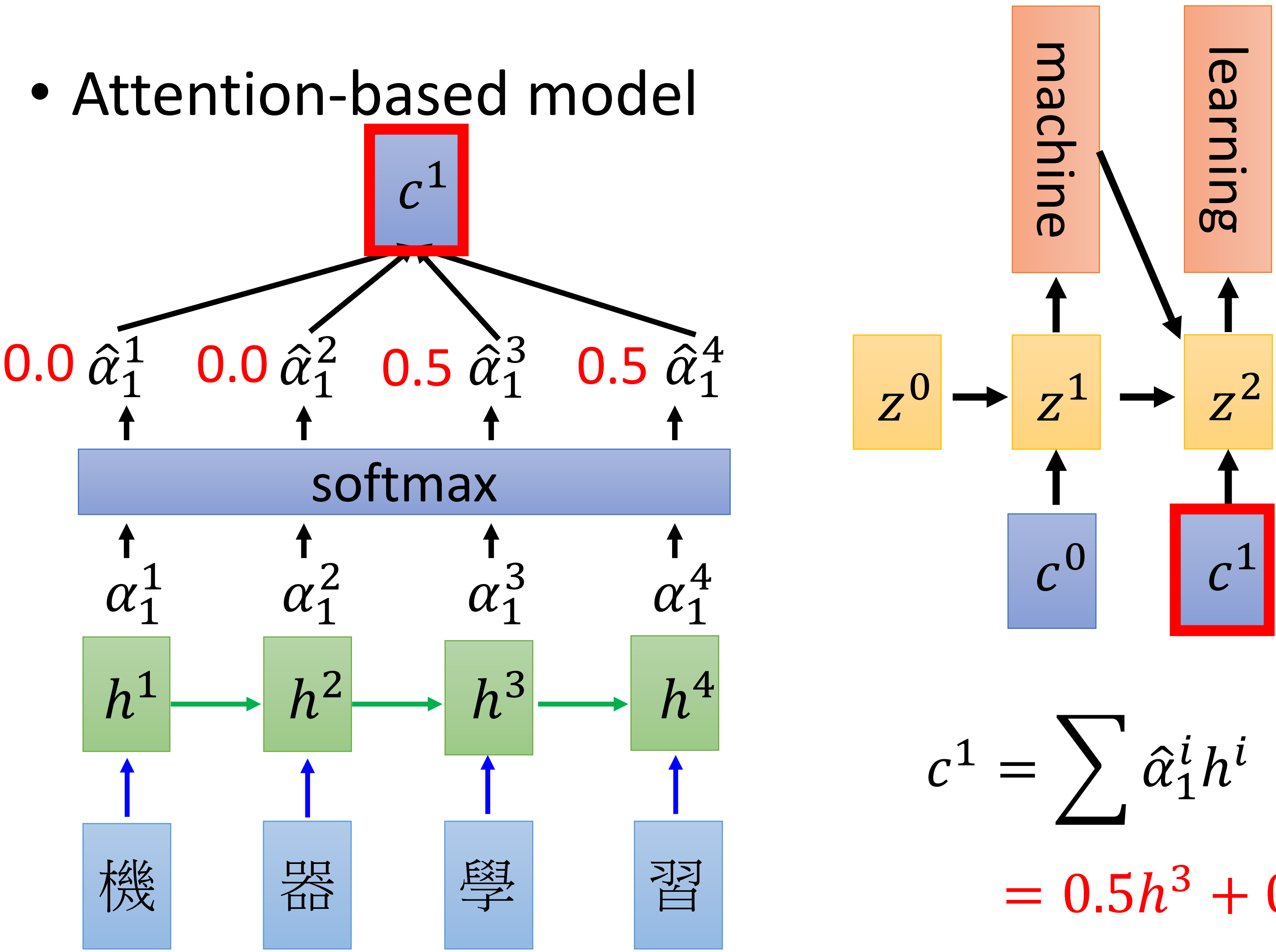
Machine Translation

- Attention-based model



Machine Translation

- Attention-based model

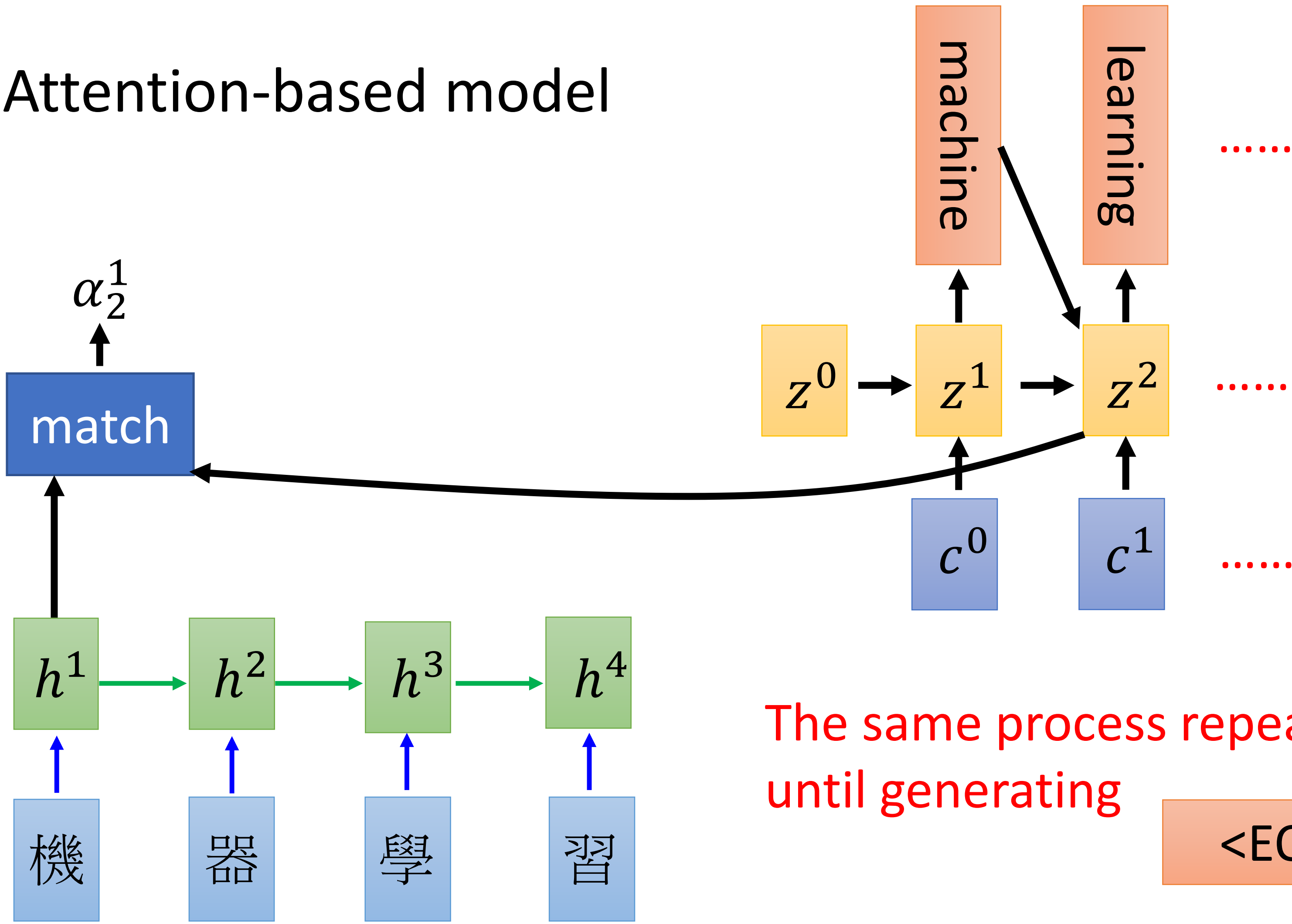


$$c^1 = \sum \hat{\alpha}_1^i h^i$$

$$= 0.5h^3 + 0.5h^4$$

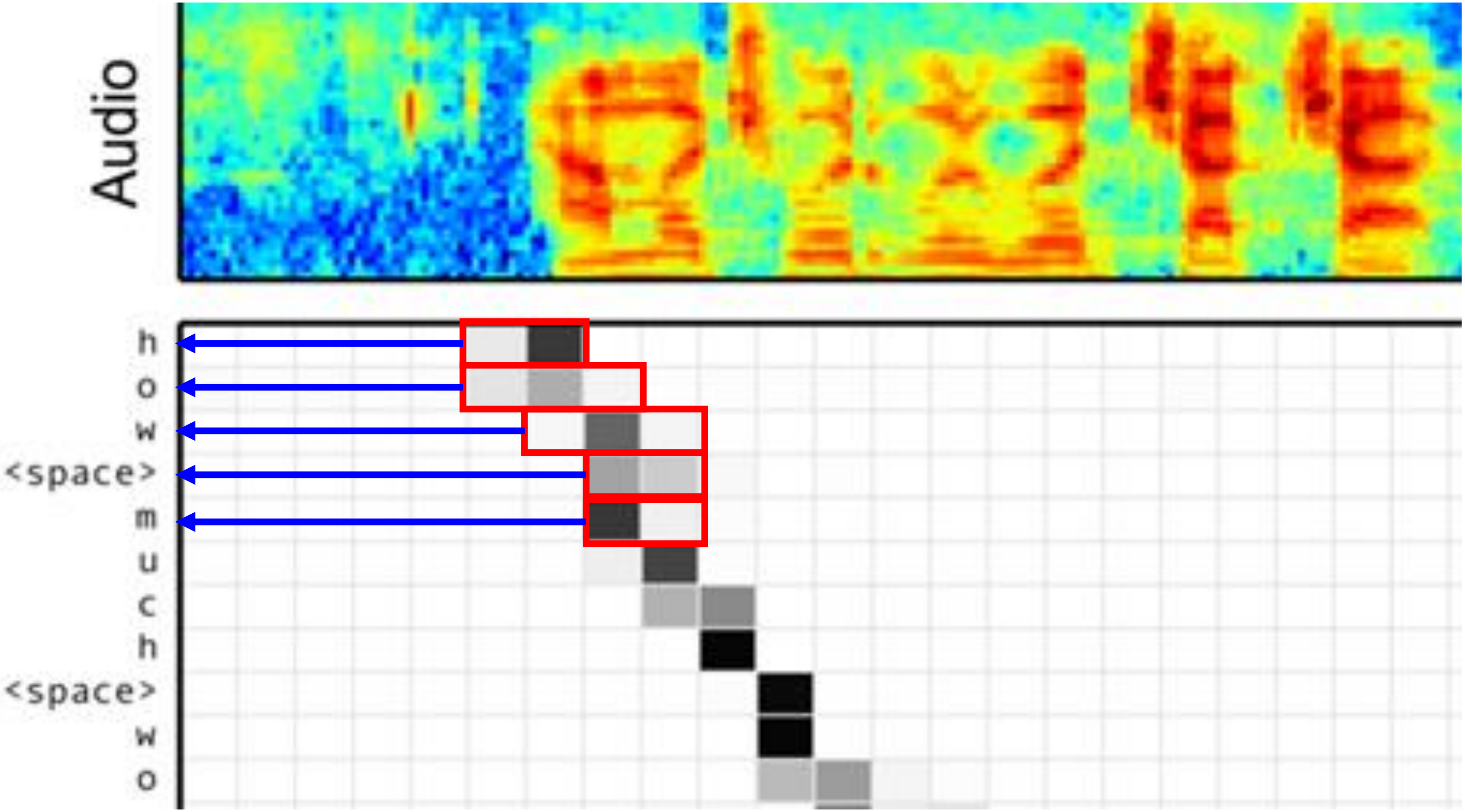
Machine Translation

- Attention-based model



The same process repeat until generating <EOS>

Speech Recognition



Model	Clean WER	Noisy WER
CLDNN-HMM [22]	8.0	8.9
LAS	14.1	16.5
LAS + LM Rescoring	10.3	12.0

William Chan, Navdeep Jaitly, Quoc V. Le, Oriol Vinyals, “Listen, Attend and Spell”, ICASSP, 2016

Image Caption Generation

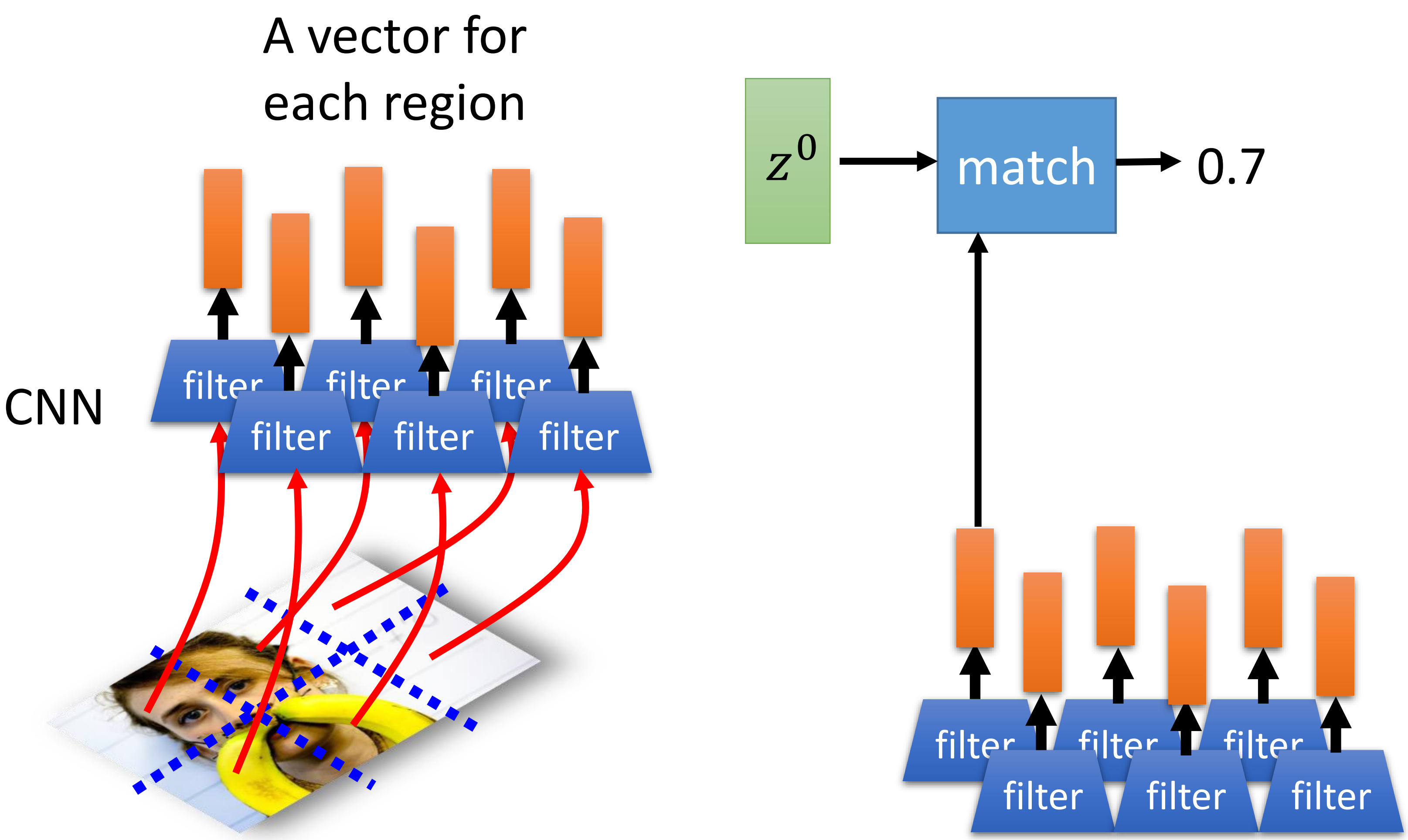


Image Caption Generation

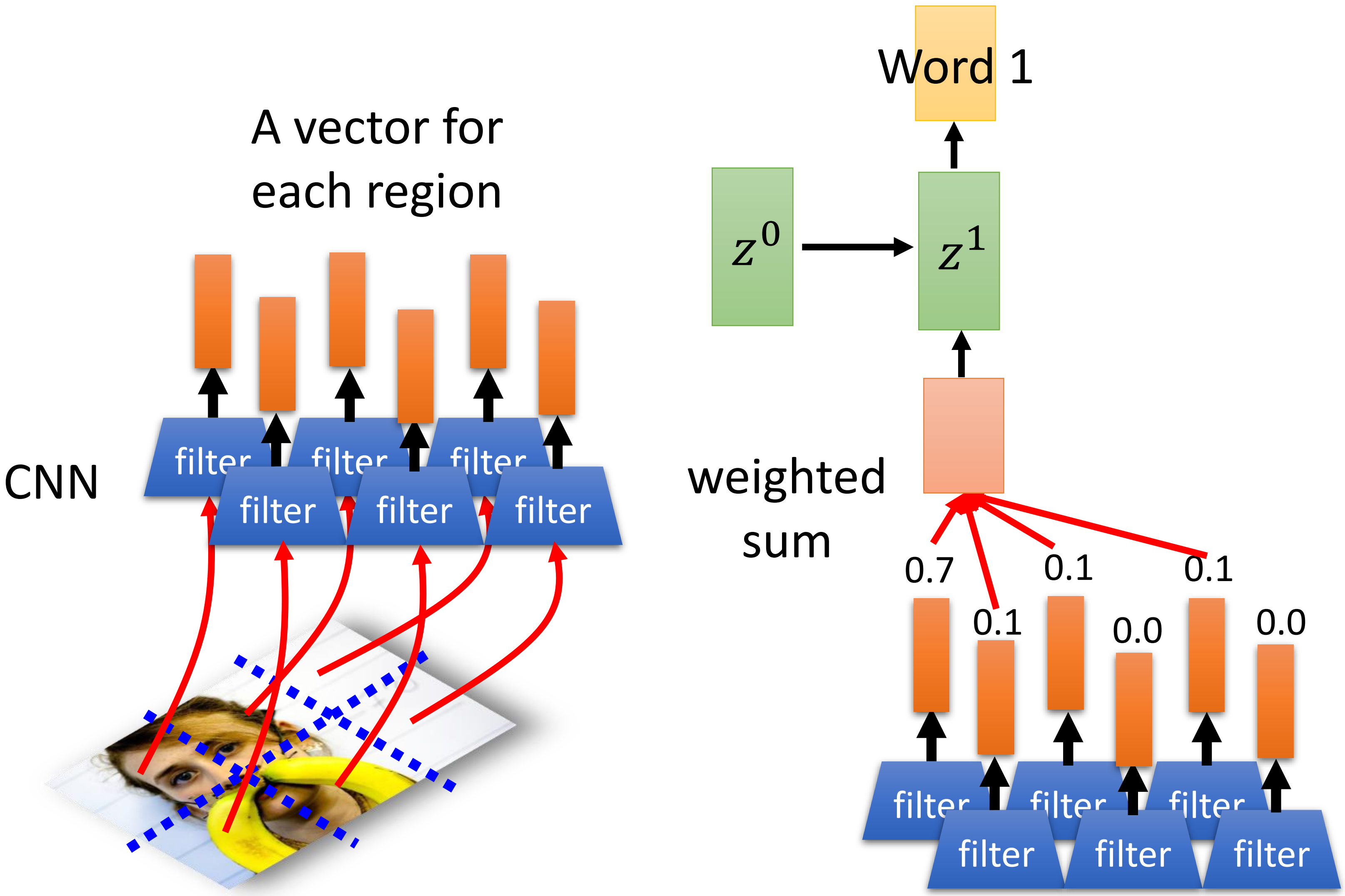


Image Caption Generation

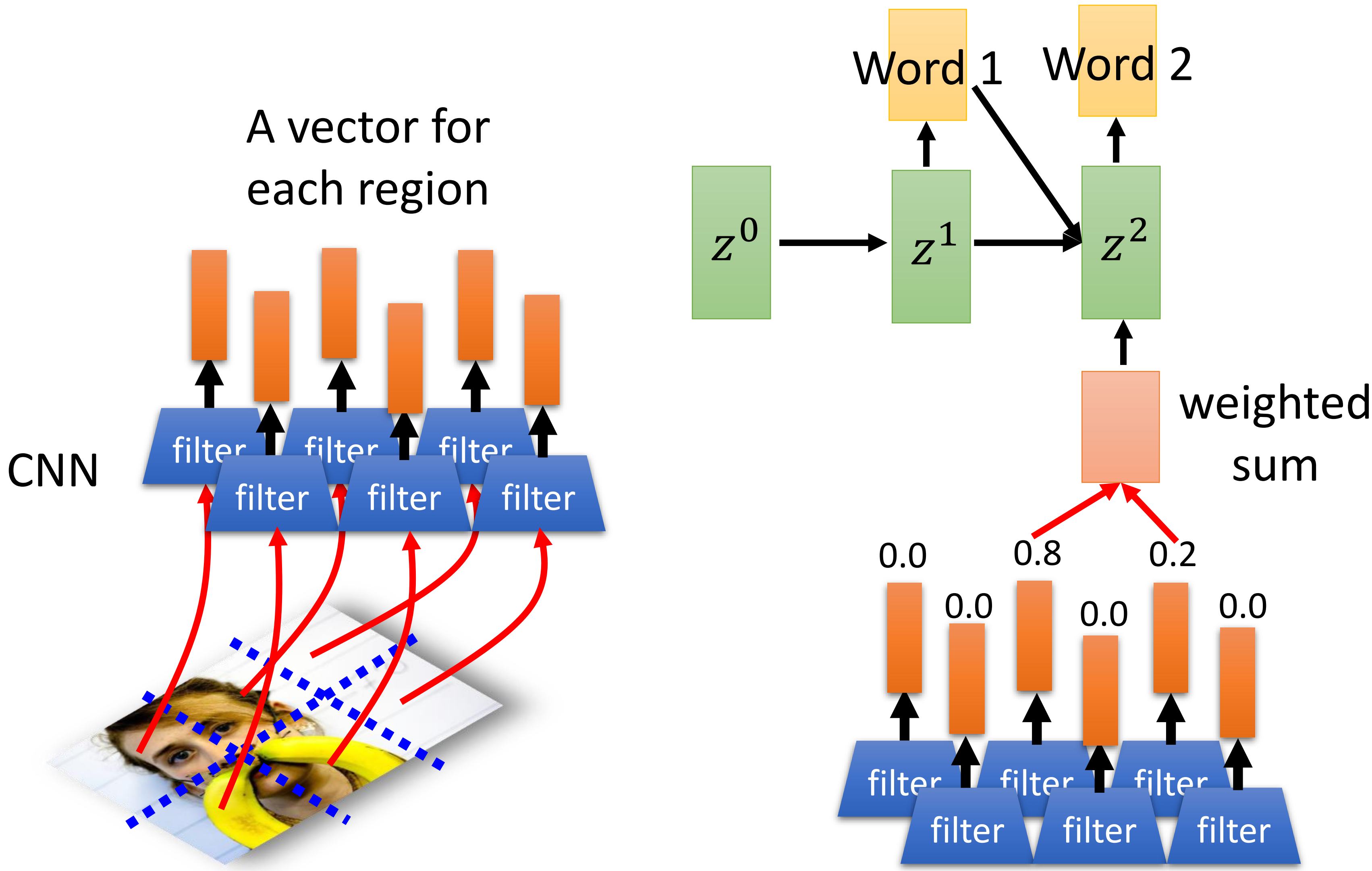


Image Caption Generation

A woman is throwing a frisbee in a park.

A dog is standing on a hardwood floor.

A stop sign is on a road with a mountain in the background.

A little girl sitting on a bed with a teddy bear.

A group of people sitting on a boat in the water.

A giraffe standing in a forest with trees in the background.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, Yoshua Bengio, “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention”, ICML, 2015

Image Caption Generation

A large white bird standing in a forest.

A woman holding a clock in her hand.

A man wearing a hat and
a hat on a skateboard.

A person is standing on a beach
with a surfboard.

A woman is sitting at a table
with a large pizza.

A man is talking on his cell phone
while another man watches.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, Yoshua Bengio, “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention”, ICML, 2015

Ref: A man and a woman ride a motorcycle

A **man** and a **woman** are **talking** on the **road**

Ref: A woman is frying food

Someone is **frying** a **fish** in a **pot**

Li Yao, Atousa Torabi, Kyunghyun Cho, Nicolas Ballas, Christopher Pal, Hugo Larochelle, Aaron Courville, “Describing Videos by Exploiting Temporal Structure”, ICCV, 2015

Tips for Generation

Attention

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, Yoshua Bengio, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", ICML, 2015

α_t^i component time

Bad Attention

$\alpha_1^1 \alpha_2^1 \alpha_3^1 \alpha_4^1$ $\alpha_1^2 \alpha_2^2 \alpha_3^2 \alpha_4^2$ $\alpha_1^3 \alpha_2^3 \alpha_3^3 \alpha_4^3$ $\alpha_1^4 \alpha_2^4 \alpha_3^4 \alpha_4^4$

w_1 w_2 (woman) w_3 w_4 (woman) …… no cooking

Good Attention: each input component has approximately the same attention weight

E.g. Regularization term: $\sum_i \left(\tau - \sum_t \alpha_t^i\right)^2$

For each component Over the generation

Mismatch between Train and Test

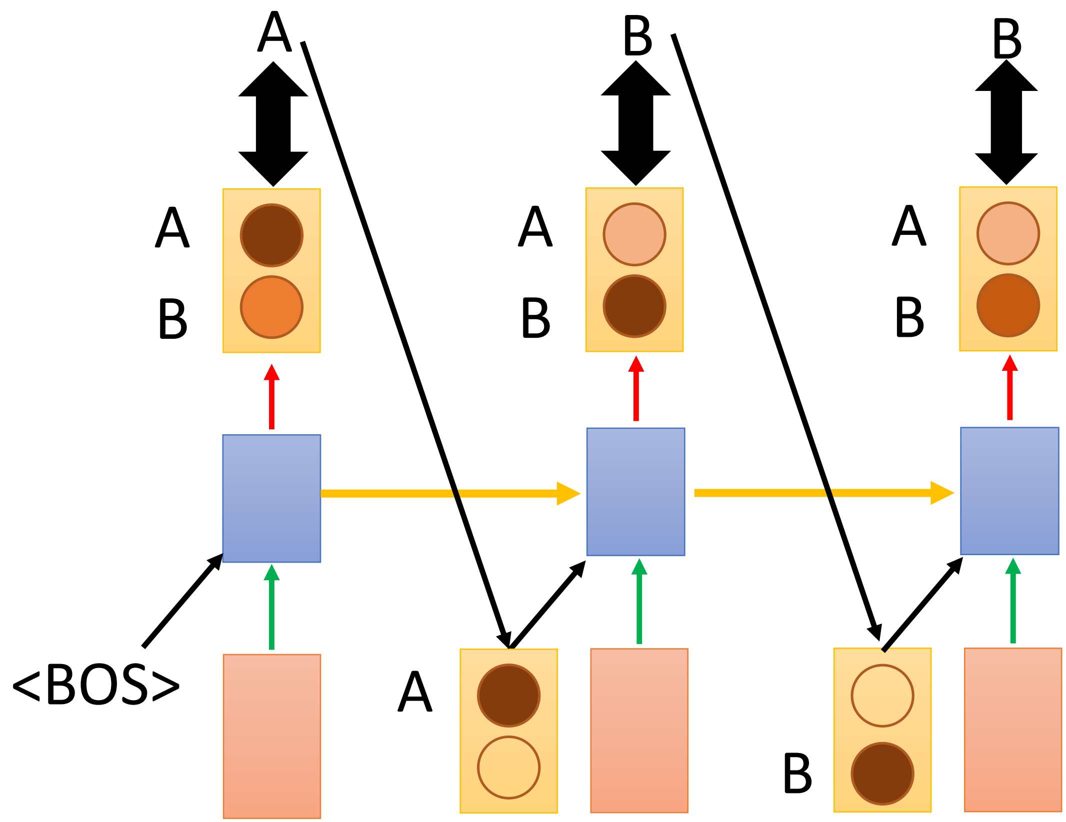
- ***Training***

$$C = \sum_t C_t$$

Minimizing cross-entropy of each component

: condition

Reference:



Mismatch between Train and Test

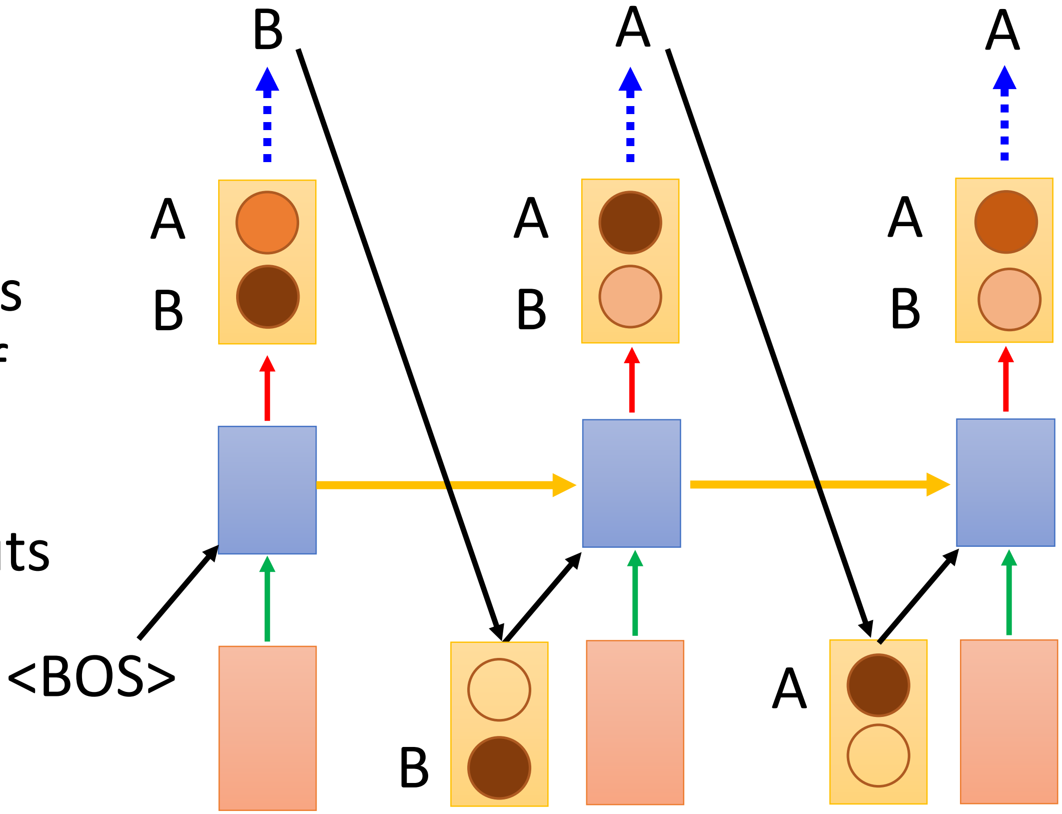
- ***Generation***

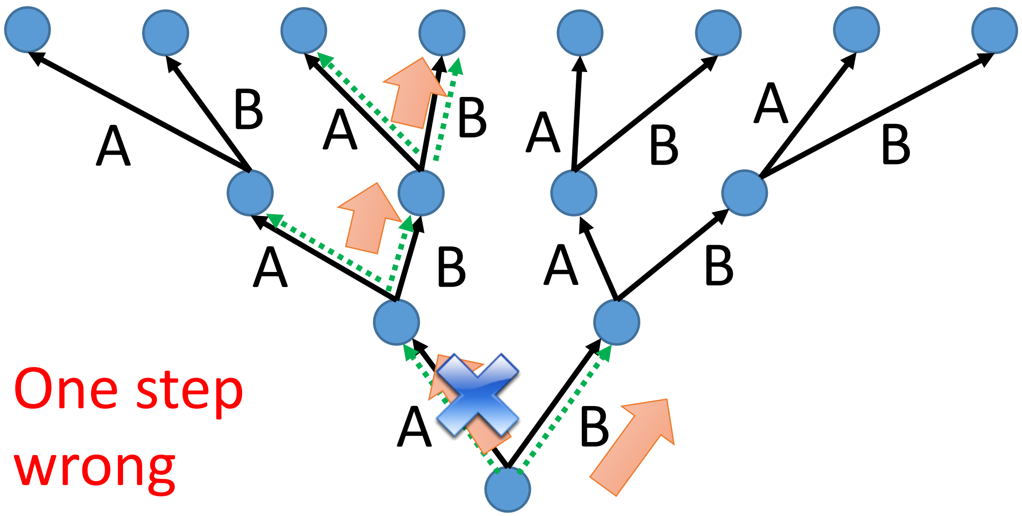
We do not know the reference

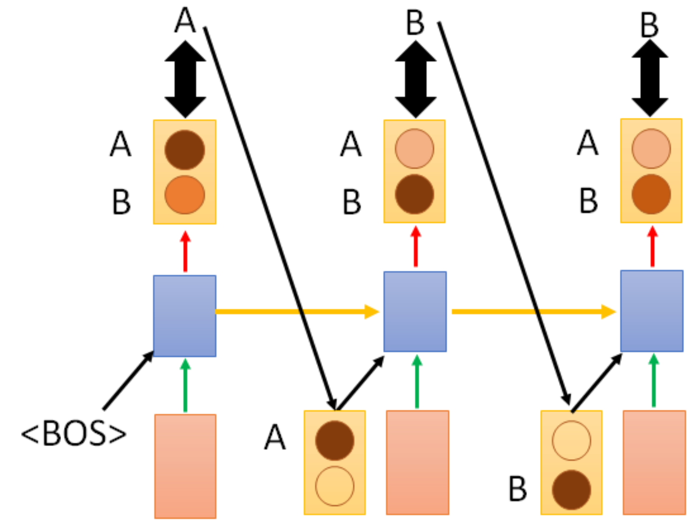
Testing: The inputs are the outputs of the last time step.

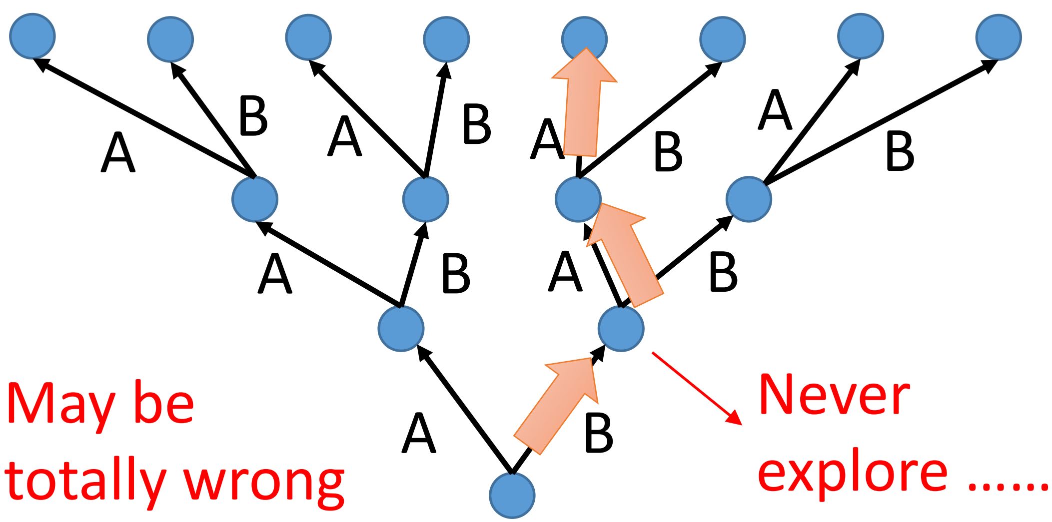
Training: The inputs are reference.

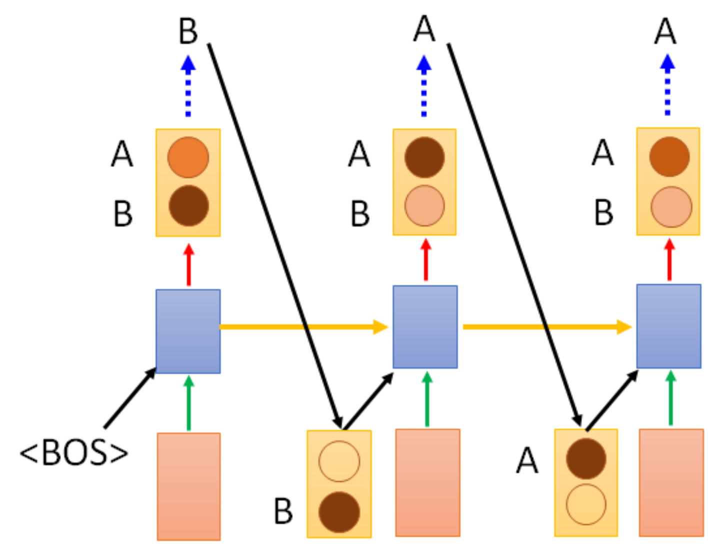
Exposure Bias




A
B
A
B
A
B
A
B
A
B
A
B
A
B
One step
wrong


A
B
B
A
B
A
B
A
B
<BOS>
A
B


A
B
A
B
A
B
A
B
A
B
A
B
A
B
May be
totally wrong
Never
explore ……


B
A
A
A
B
A
B
A
B
<BOS>
B
A

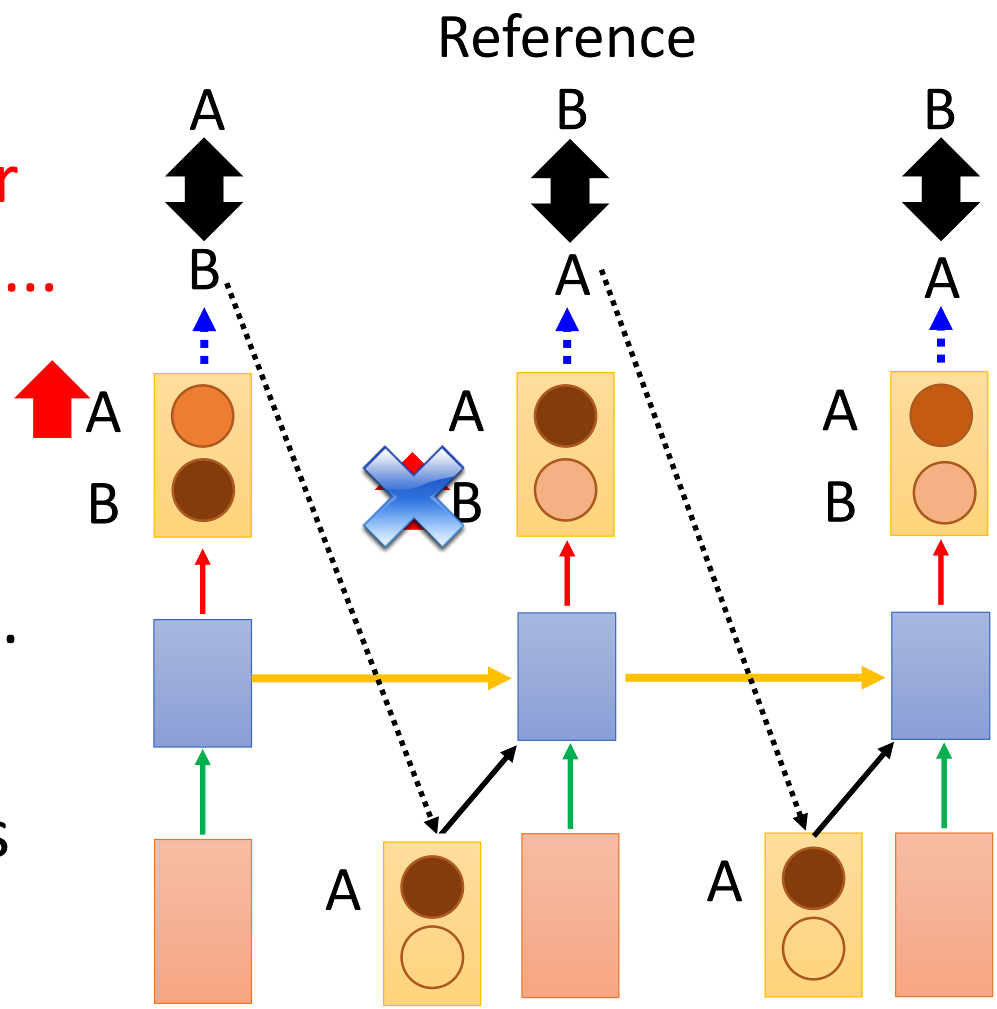
一步錯，步步錯

Modifying Training Process?

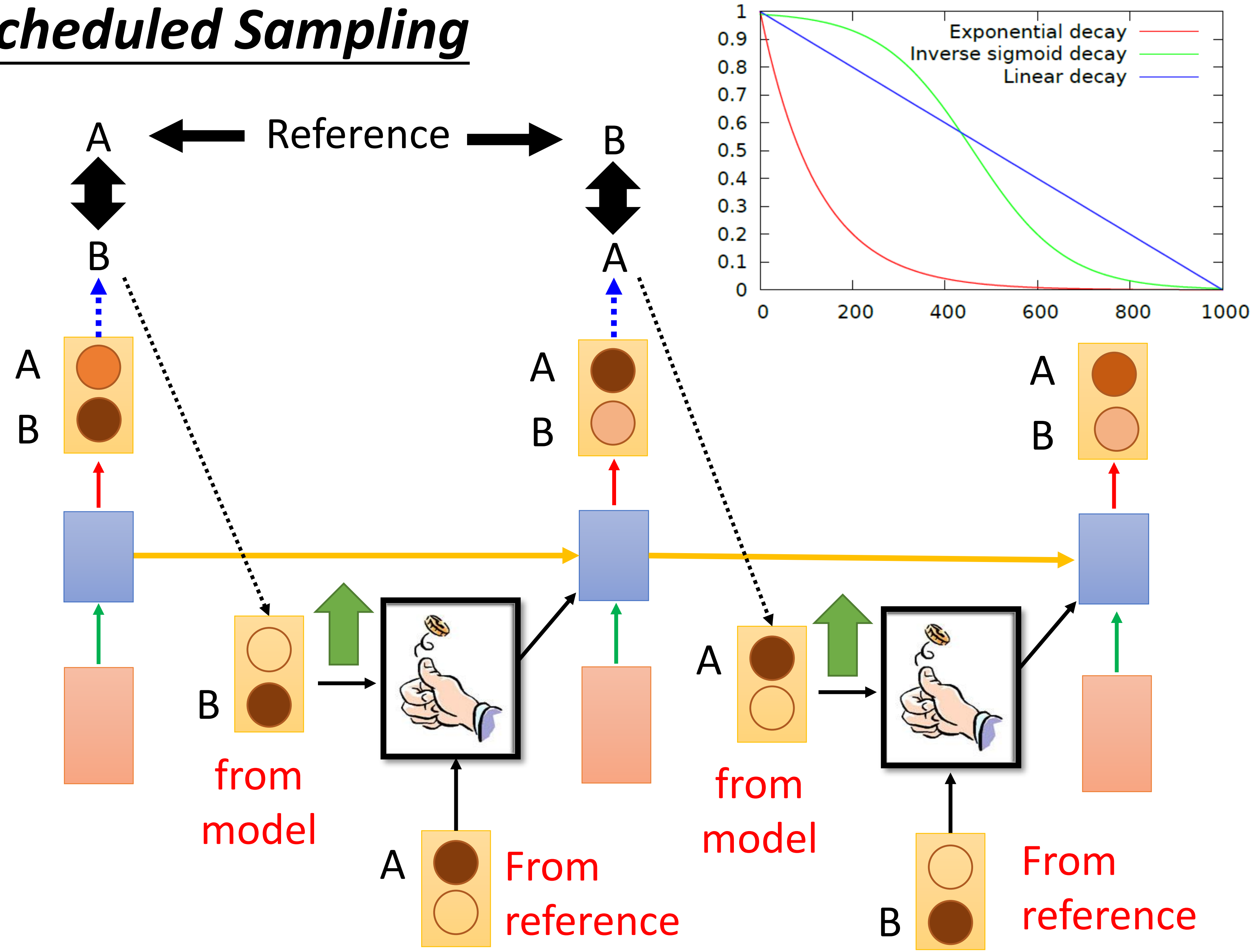
When we try to decrease the loss for both steps 1 and 2 .....

Training is matched to testing.

In practice, it is hard to train in this way.



Scheduled Sampling



Scheduled Sampling

- Caption generation on MSCOCO

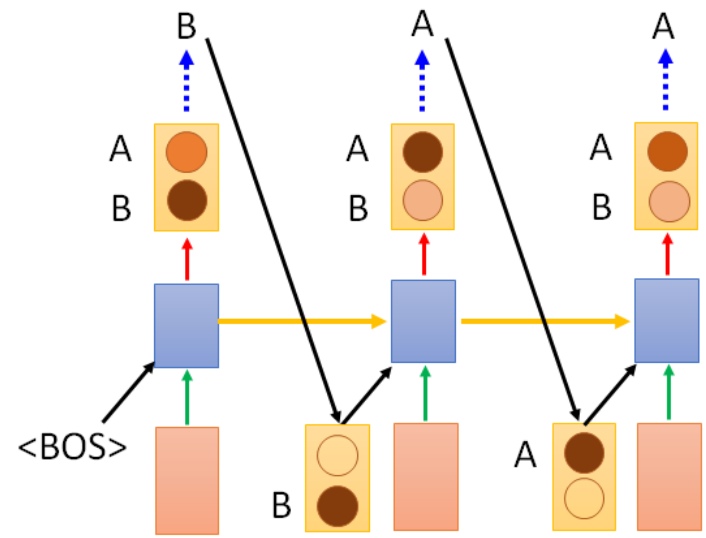
	BLEU-4	METEOR	CIDER
Always from reference	28.8	24.2	89.5
Always from model	11.2	15.7	49.7
Scheduled Sampling	30.6	24.3	92.1

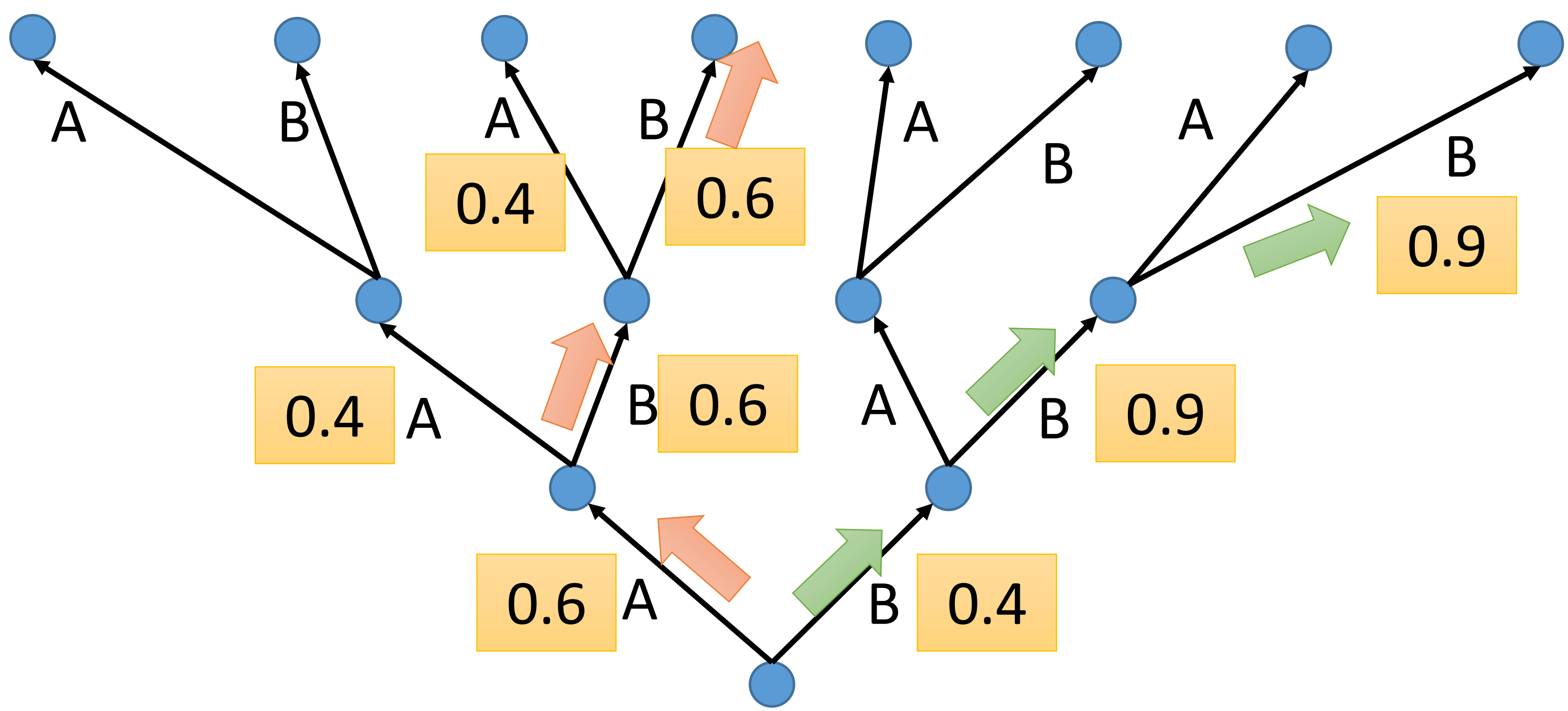
Samy Bengio, Oriol Vinyals, Navdeep Jaitly, Noam Shazeer, Scheduled Sampling for Sequence Prediction with Recurrent Neural Networks, arXiv preprint, 2015

Beam Search

The green path has higher score.

Not possible to check all the paths

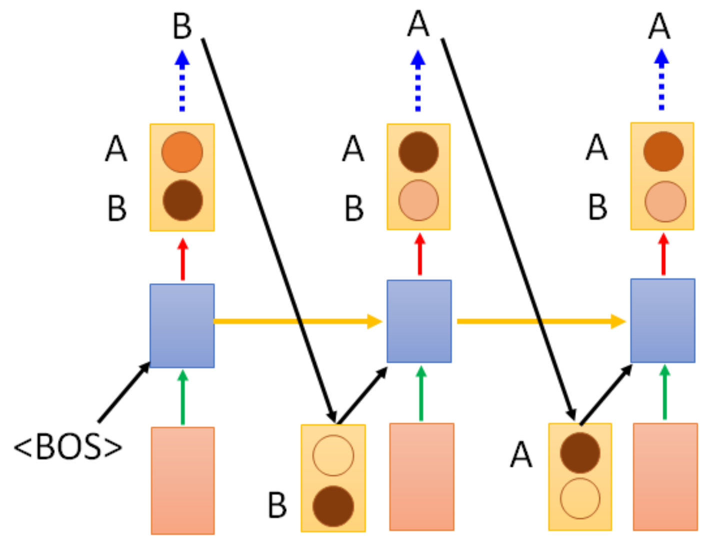


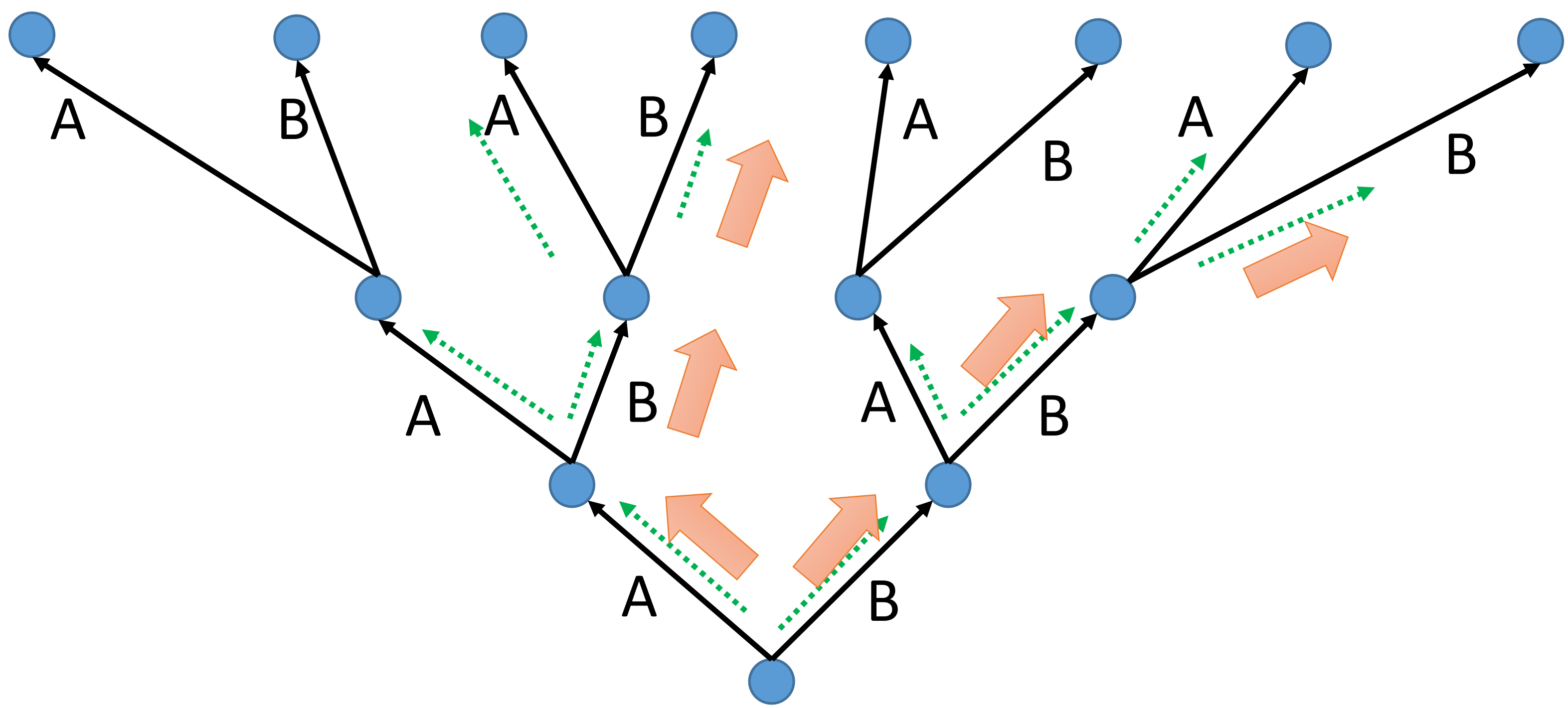


Beam Search

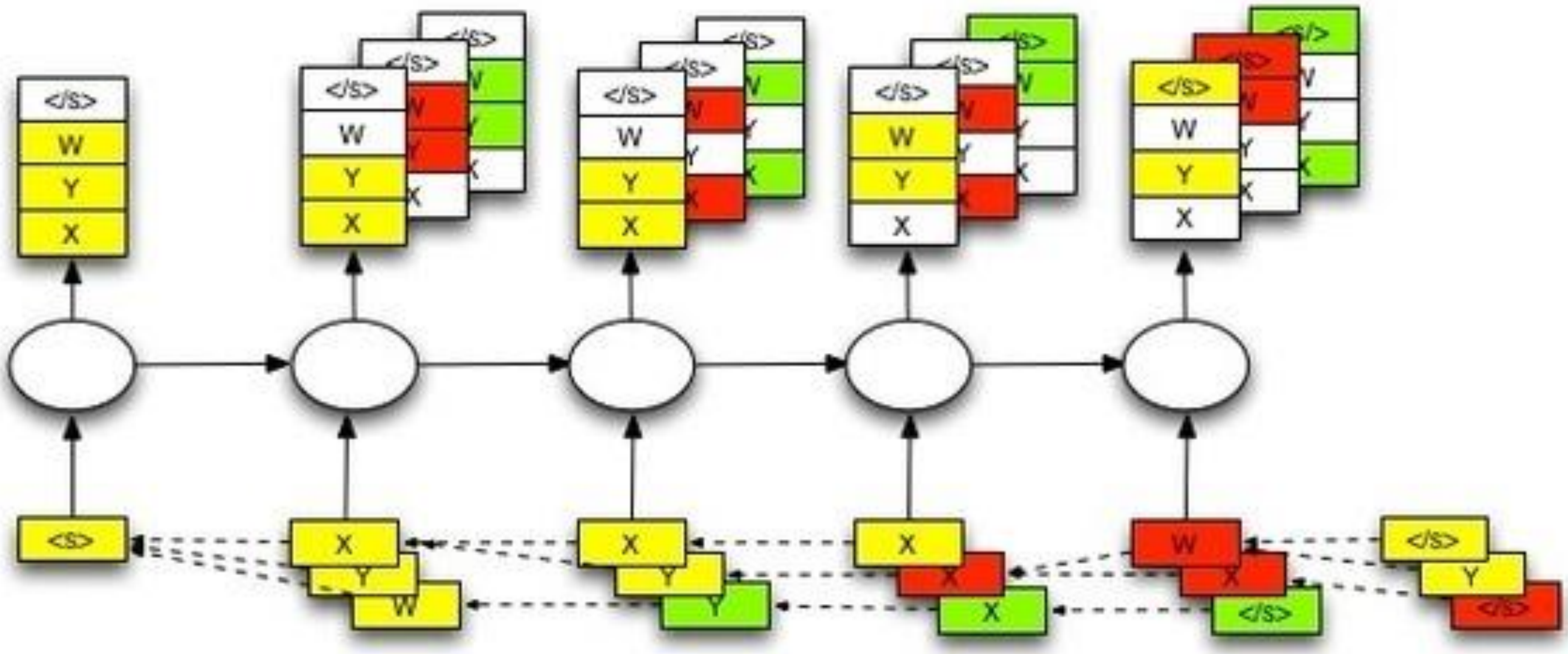
Keep several best path at each step

Beam size = 2





Beam Search



The size of beam is 3 in this example.

https://github.com/tensorflow/tensorflow/issues/654#issuecomment-169009989

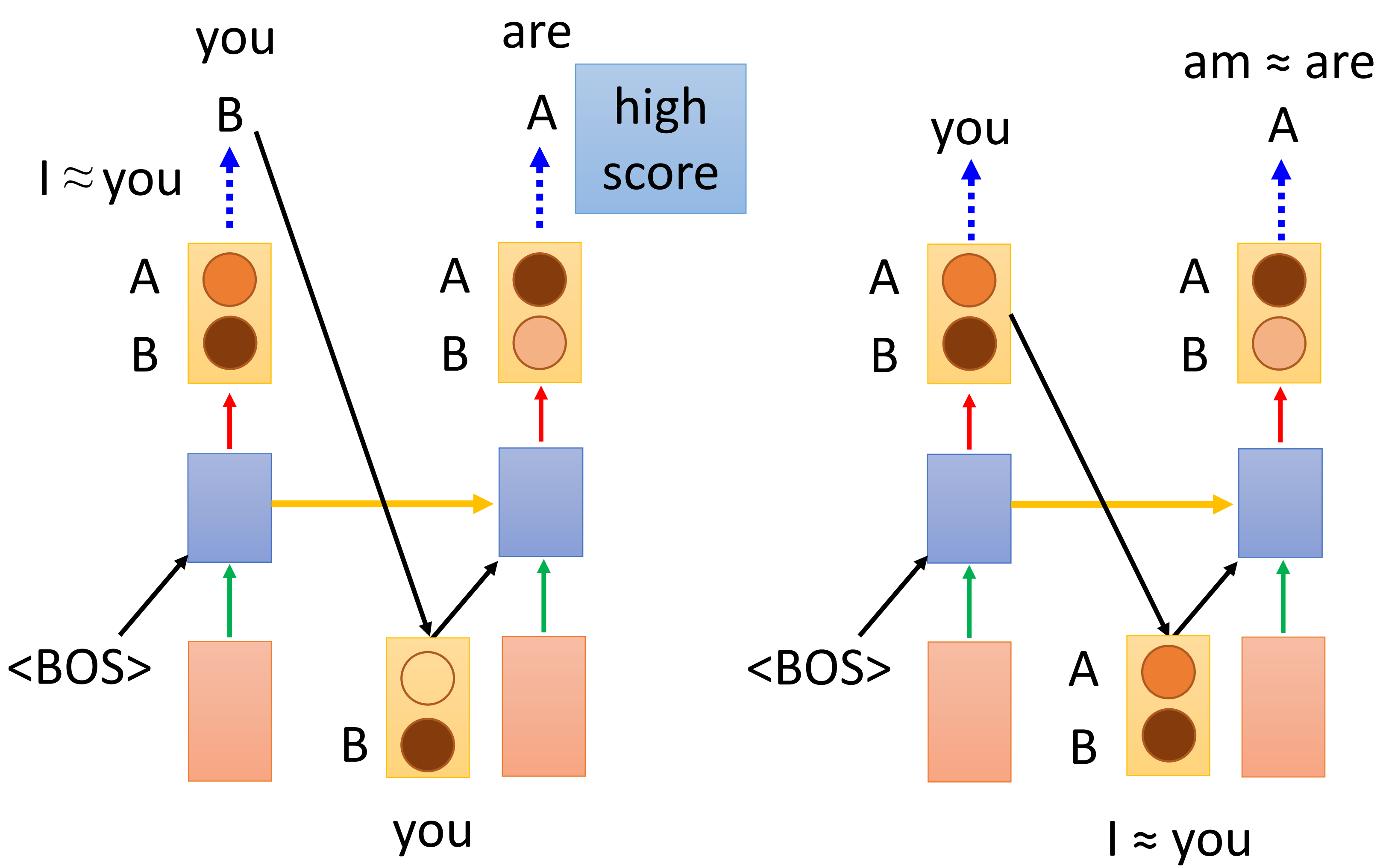
Better Idea?

I am ……

You are ……

I are ……

You am ……



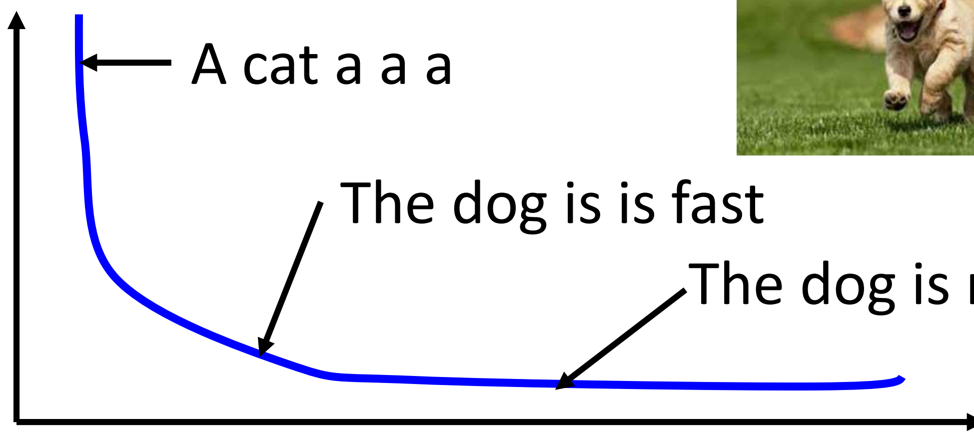
Object level v.s. Component level

- Minimizing the error defined on component level is not equivalent to improving the generated objects

Ref: The dog is running fast

$$C = \sum_t C_t$$

Cross-entropy of each step

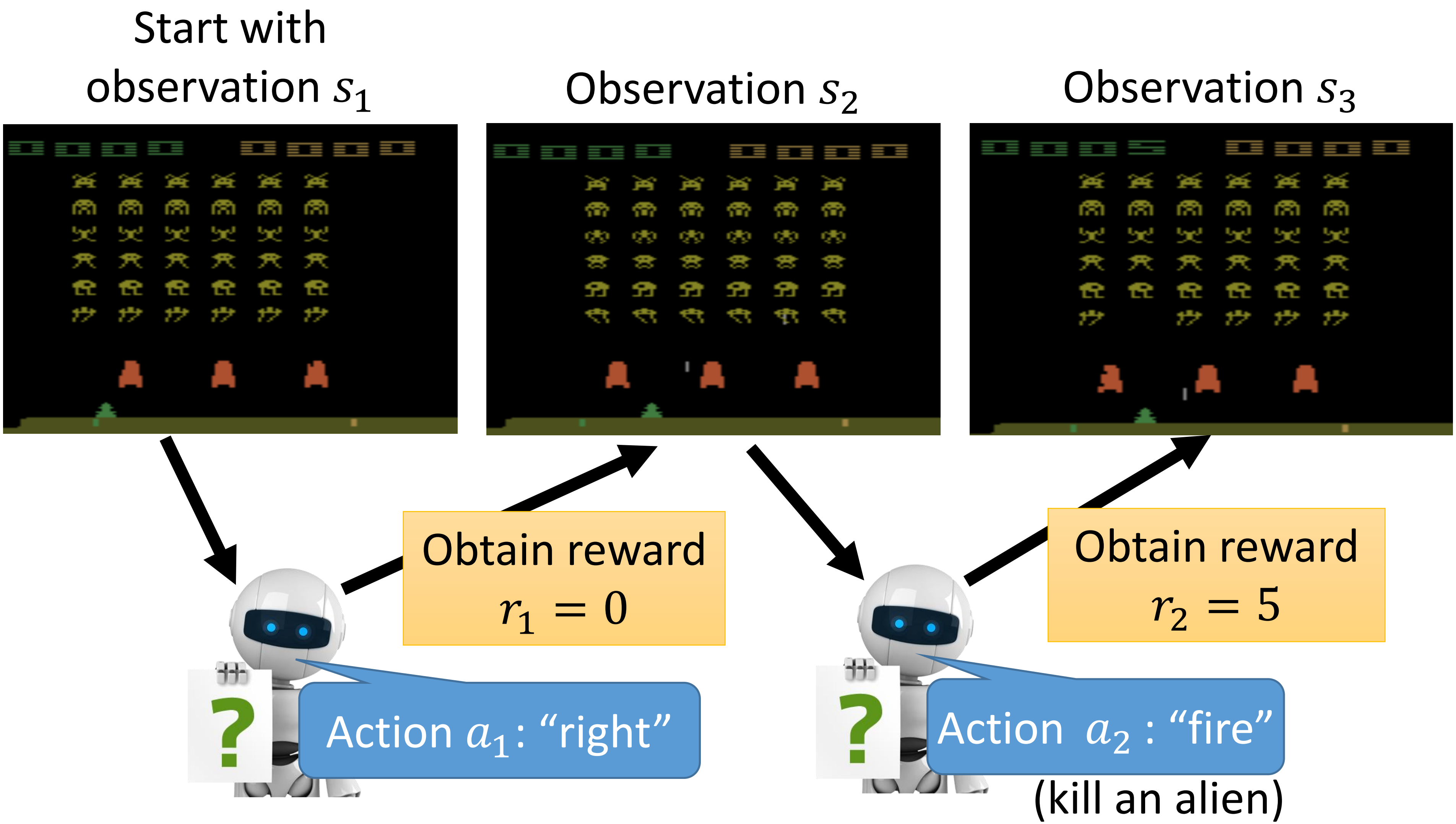


Optimize object-level criterion instead of component-level cross-entropy. object-level criterion: $R(y, \hat{y})$

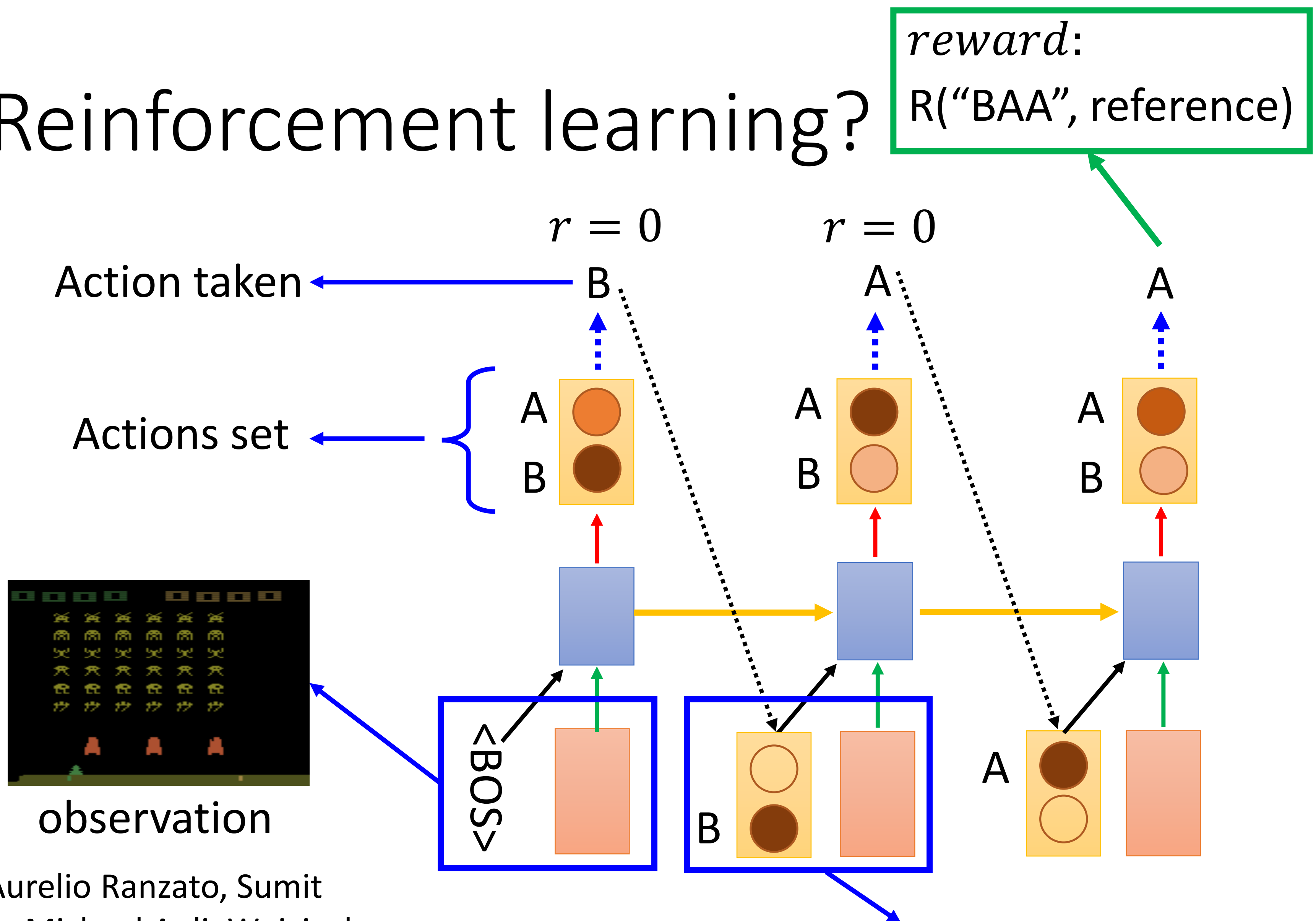
Gradient Descent?

y: generated utterance, $\hat{y}$: ground truth

Reinforcement learning?



Reinforcement learning?



The action we take influence the observation in the next step

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, Wojciech Zaremba, "Sequence Level Training with Recurrent Neural Networks", ICLR, 2016

Concluding Remarks

- RNN with Gated Mechanism
- Sequence Generation
- Conditional Sequence Generation
- Tips for Generation